Call-A.P.P.L.E.™

Volume VI, Number 3 March 1983 $2.50 ($3.00 Canada, £1.75 UK.)

EXPLORING THE WORLD OF APPLE:

Switchcraft: Memory Management

Page 9

A CALL —A.P.P.L.E. REVIEW:

Transforth

Page 13

Anatomy of an Eamon Adventure

Page 21

A CALL —A.P.P.L.E. REVIEW:

Ampersand Utilities

Page 37

Apple PugetSound Program Library Exchange

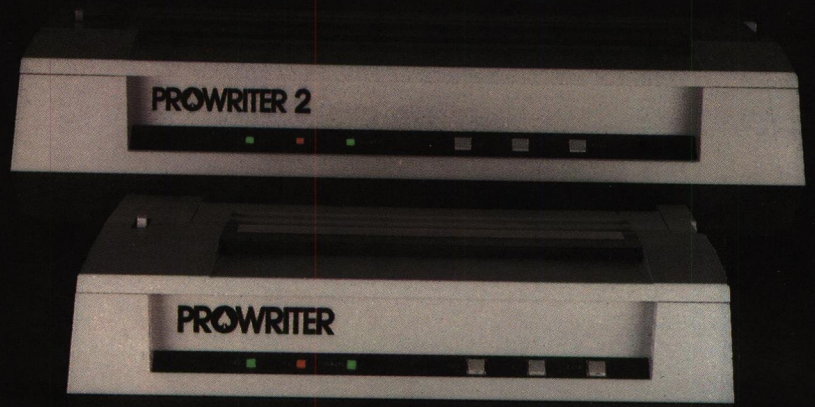
THE LEADING EDGE IN PRINTERS

ONE GREAT LINE. ONE GREAT WARRANTY.

Finally, there's one full family of printers that covers every business or word processing application—all from C. Itoh, a company known for packing more product into less price; and all distributed exclusively by Leading Edge, a company known for searching out and providing that very thing. Which means that one call to one source can get you any printer, any time you need it, for any purpose. All backed by a full years' warranty from Leading Edge. (Try *that* on any other line of printers.)

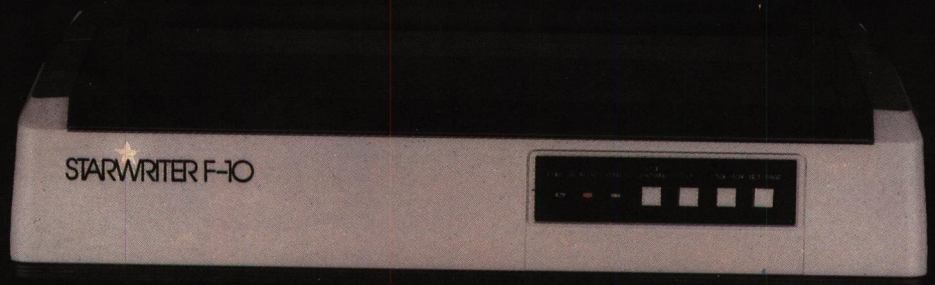
THE PRO'S.

The Prowriters: business printers—and more. The "more" is a dot-matrix process with more dots. It gives you denser, correspondence quality copy (as opposed to business quality copy, which looks like a bad job of spray-painting).
Prowriter: 120 cps. 80 columns dot matrix compressable to 136. 10" carriage. Parallel or serial interface.
Prowriter 2: Same as Prowriter, except 15" carriage allows full 136 columns in normal print mode. Parallel or serial interface.



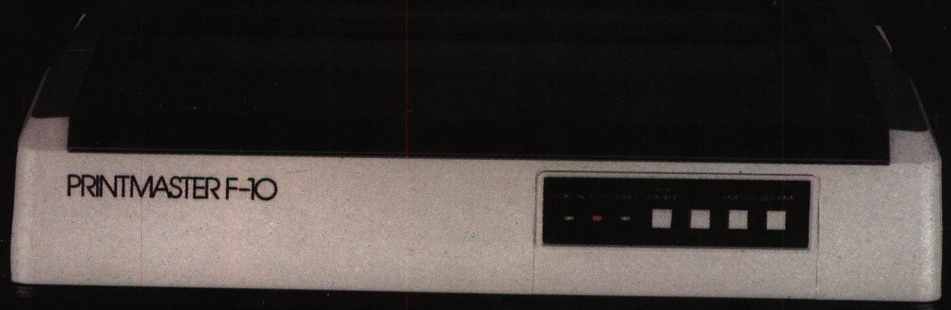
THE STAR.

The Starwriter F-10. In short (or more precisely, in a sleek 6" high, 30-pound unit), it gives you more of just about everything—except bulk and noise—than any other printer in its price range. It's a 40 cps letter-quality daisy-wheel with a bunch of built-in functions to simplify and speed up word processing. It plugs into almost any micro on the market, serial or parallel.



THE MASTER.

The Printmaster F-10. Does all the same good stuff as the Starwriter except, at 55 cps, the Master does it faster.



Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021. Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.

Circle 75 on Reader Service Card

EVERYONE NEEDS A . . .

UNINTERRUPTABLE POWER SOURCE

ADVANCED DESIGN
NEW RELIABILITY
GUARDIAN PROTECTS
ELECTRONIC SAFETY
LIFESAVER FOR DATA

$595

SAVE YOUR DATA FROM POWER OUTAGES!

BACKUP FOR YOUR COMPUTER, MONITOR, PRINTER AND 5¼" FLOPPY AND HARD DISC DRIVE

• Automatically stops annoying problems from power line interruptions and brown outs • You need standby power to save data • Maintenance free backup power available in 115 volt or 220 volt • 50 or 60 HZ • 150 watts • Complete versatility — operate your system from a 12 volt source, i.e., automobile cigarette lighter, boat or airplane • Rugged self contained gel cell battery • No voiding warranty — no cutting wires • Automatic audio alarm warning tone during commercial power failure or interrupt • UL listed • FCC approved • Transient voltage suppressor gives added insurance from line voltage spikes, utilizing Zener Ray™ • Green/red LED power status indicator • Green — normal AC line power • Slow blinking red — at least 6 minutes of remaining standby power • Fast blinking red — approximately 2 minutes of remaining battery power • Solid state technology unexcelled by any UPS power unit in its class.

RH ELECTRONICS, INC.

COPYRIGHT © 1981 - PATENTS PENDING

566 IRELAN, BUELLTON, CA 93427

(805) 688-2047

Circle 60 on Reader Service Card

SEE YOUR RH ELECTRONICS PRODUCTS DEALER

FOR YOUR APPLE II*:

SUPER FAN II™	$ 74.95
SUPER FAN II™/ZENER RAY™	$109.00
SUPER RAM II™	$125.00
RH 12 VOLT TRANSVERTER	$149.00

FOR MICRO COMPUTERS:

GUARDIAN ANGEL™	$595.00

APPLE PUGETSOUND PROGRAM LIBRARY EXCHANGE:

CLUB INFORMATION

GENERAL OFFICES
21246 68th Ave. S.
Kent, WA 98032 — (206) 872-2245
Hours: 9 - 4 Monday - Friday
Members Services/Dealer Orders

EDITORIAL DEPARTMENT
5606 California Ave. S.W.
Seattle, WA 98136—(206)932-8288
Hours: 8-4 Monday - Friday

ADVERTISING REPRESENTATIVES
Grawin Publications
1020 Lloyd Building
Seattle, WA 98101 (206) 223-0861

MEETING NOTICES

7:30 PM, Tuesday, March 15, 1983
New A.P.P.L.E. Office
21246 68th Ave. S., Kent
Guest Speaker
Jonathan V. Post
"Birth of the Bio Computer"

7:30 PM, Tuesday, April 19, 1983
A.P.P.L.E. Office
21246 68th Ave. S., Kent
Speaker to be announced

7:30 PM. Tuesday, May 17, 1983
A.P.P.L.E. Office
21246 68th Ave. S., Kent
Speaker to be announced

Call —A.P.P.L.E. HOTLINE

Technical/programming assistance only

DAYTIME HOTLINE: (206) 872-9004
9 AM - 3 PM Only
Mon. - Fri. Only

EVENING HOTLINE: (206) 932-6588
6 PM - 10 PM
7 days per week

WEEKEND HOTLINE: (206) 932-6588
9 AM - 3 PM
6 PM - 10 PM
Sat. - Sun.

All times shown are PACIFIC time.

Membership Information

Please direct telephone questions on membership and orders to (206) 271-4514.

One Time Apple—Cation Fee	$25.00
Annual Dues	$20.00
Total	$45.00

Additional postage required for foreign memberships and shipments.

Additional membership postage:

- Canada, West Indies, Central America, Columbia, Venezuela and Mexico — Add $15.00 ($60.00 total)
- All S. American countries (except above), Europe and N. Africa — Add $21.00 ($66.00 total)
- Asia, Mid East, S. Africa, Australia and Pacific Islands — Add $27.00 ($72.00 total)

All orders must be prepaid in U.S. funds only.

Membership is on an annual, 12 month basis, commencing with the month of enrollment. Upon receipt of total membership fees, new members will receive an Apple—Cation blank, order form and description sheet of club products.

Please allow up to 30 days for a turnaround on membership and orders.

Call —A.P.P.L.E. is mailed free twelve time per year as benefit of membership. Subscriptions are not available.

1983 back issues available at $2.50 each.

Call —A.P.P.L.E. bound volumes for previous years are available as indicated on the member order form on the last page.

Institutional memberships are now available. Contact General Office for details.

Prospective authors please send self-addressed stamped envelope (37¢ postage) to Editorial Dept. for a copy of Author's guidelines.

Cover and interior design by Vic Warren Design.

IMPORTANT NOTICE

Please see our new address and telephone number for the GENERAL OFFICE at above left. This change became effective February 1, 1983.

The move to our new quarters also allows us to expand the capacity of our Hotline service. Weekdays from 9 am - 3 pm a staff of trained technical specialists will be on hand at (206) 872-9004.

Evenings from 6 pm - 10 pm and weekends, 9-3 and 6-10, service remains at the old number, (206) 932-6588.

Changes of address must be received at the General Office at least 30 days prior to effective date of move. Issues missed due to non-receipt of change of address notification must be separately reordered by the member at current cost. Change of address form may be found on page 127 of this issue.

A.P.P.L.E. CRATE BULLETIN BOARD
(206) 935-9119
(usually 24 hours)

Officers and Directors

Dick Hubert, President	(206) 271-6939
Bob Huelsdonk, Vice President	(206) 362-4910
Dave Lingwood, Secretary	(206) 244-9360
Bob Clardy, Treasurer	(206) 641-1917
Val J. Golding, Director	(206) 932-6588
Ken Smith, Director	(206) 322-4199
Don Williams, Director	(206) 242-6807

Office Staff

Dick Hubert	Executive Director
Dave O'Leary	Administrative Assistant
Sally Davies	Admin. of Information Systems
Arliss Sword	Dealer Orders
Patti Hoffmann	Special Projects Coordinator
Gunter Hirt	Bookkeeping, Member Records
Joff Morgan	Technical Operations
Cheryle McCalmant	Order Processing
Kris Mortenson	Order Processing
Sharon Birdsell	Order Processing
Gayle Rank	Secretary/Receptionist
Mike Christensen	Systems Development
James Redhead	Receiving
Brad Phillips	Shipping/Receiving
Steve Johnson	Ap•tronics
Steve Ross	New Product Development
Garry Reinhardt	Systems Development
Butch Greathouse	Technical Support

Magazine Staff

Val J. Golding	Editor	(206) 932-8288
Kathryn Hallgrimson	Associate Editor	(206) 932-8288
Kathy Pride	Editorial Assistant	(206) 932-8288
Jonathan V. Post	Contributing Editor	(206) 824-7641
Bruce van Norman	Pascal Editor	(206) 932-8288
Kevin O. Donohoe	Production Editor	(206) 872-2245
Jim Hockenhull	Staff Writer	(509) 334-4137
Clifton M. Howard, M.D.	Staff Writer	(201) 768-9274
Paul Mosher, M.D.	Staff Writer	(518) 765-4118
Bill Parker	Staff Writer	(619) 297-9343
Lee Reynolds	Staff Writer	(305) 722-2639
Clay Ruth	Staff Writer	(219) 865-6165
David G. Sparks	Staff Writer	(319) 232-2313
Ralph H. Swerdlow, M.D.	Staff Writer	(916) 920-8709
Grawin Publications	Production, Advertising	(206) 223-0861

A.P.P.L.E. consultants will be found on page 127 of this issue.

Call-A.P.P.L.E.™

Volume VI, Number 3 March 1983/Menu

Val J. Golding, Editor
Kathryn Hallgrimson Suther, Associate Editor

ENTRÉES

9 EXPLORING THE WORLD OF APPLE:
Switchcraft: Memory Management
Coleman Logan

13 A CALL —A.P.P.L.E. REVIEW:
Transforth
Charles Wells/Reviewer

21 **Anatomy of an Eamon Adventure**
Robert Plamondon

37 A CALL —A.P.P.L.E. REVIEW:
Ampersand Utilities
David Sparks/Call —A.P.P.L.E. Staff Writer

À LA CARTE

47 A CALL —A.P.P.L.E. PERSONALITY:
A Visit With Cornelis Bongers
Ralph Swerdlow, M.D./Call —A.P.P.L.E. Staff Writer

51 **Escape From the Motherboard**
Donald W. Miller, M.D.

53 **APPLE — The Magician**
Luis Hernandez

59 A CALL —A.P.P.L.E. TUTORIAL:
Error Message Dictionary
Lee Webster

67 A CALL —A.P.P.L.E. TECHNIQUE:
An Alternative Text Page
Glen Bredon

71 A CALL —A.P.P.L.E. REVIEW:
Apple Flasher
Walter Hopkins/Reviewer

75 FROM THE VERY CORE OF APPLE:
Wait-ing in the Monitor
John K. Elberfeld

77 **Waiting for a Keypress**
Ken Kashmarek

78 **Microsolitude**
Mike Shadick

79 PASCAL GRAPHICS:
Seeing Double Safely
Lawrence O'Glasser

81 A CALL —A.P.P.L.E. UTILITY:
CP/Muffin
Val J. Golding

89 JEANNE'S PROCEDURES:
A Beginner's Guide to the Pascal Editor
Terry Reilly

93 A CALL —A.P.P.L.E. TECHNIQUE:
Customizing GPLE Escape Tables
Edward Nutter

99 **Making It Last**
Robert Hanson

105 A CALL —A.P.P.L.E. TECHNIQUE:
Converting "NEW" to "FP" Type Files
W. T. Doyle

107 **Delete That Incubus!**
Howard S. Balsam

111 A CALL —A.P.P.L.E. TECHNIQUE:
FIX APCPM
Murray Arnow

113 **The Word**
Mike Shadick

114 **Beep: A Tale of (T)ERROR**
Christopher Volpe

115 **Robotics**
Michael Thyng/Call A.P.P.L.E. Staff Writer

HORS d'OEUVRES

5 The Editor Bytes Back
The Editor

5 BOOT
Craig MacIntosh

7 Bytes From A.P.P.L.E.
Val J. Golding

118 Tomorrow's Apples Today
Kevin O. Donohoe/Editor

127 Call —A.P.P.L.E. Consultants

127 A.P.P.L.E. Order Form

128 Call —A.P.P.L.E. Unclassifieds

Apple user groups may reprint without prior permission any portion of the contents herein, provided author, title, and publication credits are given and a written notice of such use furnished to the editorial office. For commerical applications of programs herein, the author should be contacted c/o General Office. Call —A.P.P.L.E. is published monthly by Apple Puget Sound Program Library Exchange, 21246 - 68th Ave. S., Kent, WA 98032. Subscriptions are not available but copies are mailed free as a benefit of A.P.P.L.E. membership.

Entire contents copyright© 1983 by

Apple PugetSound Program Library Exchange

NEW!

SCOOTER™

HIGH GRADE COMPONENTS FROM OHM/ELECTRONICS

Now your computer dealer can offer the quality and selection electronics professionals demand.

CABLES & CORDS • RIBBON CABLE ASSEMBLIES
CAPACITORS • CONNECTORS • FUSES & HOLDERS
ELECTRONIC HARDWARE • HEAT SINKS
INDICATOR LIGHTS • INTEGRATED CIRCUITS
IC SOCKETS • KEYBOARDS • LED'S • OUTLET STRIPS
POTS & CONTROLS • RESISTORS • SEMICONDUCTORS
SOLDER • SWITCHES • TEST LEADS

Scooter brings you the best known names in electronics including Amphenol, Allen-Bradley, Bourns, Grayhill, Motorola, Panasonic, Sprague and TRW-Cinch. And they're all at your computer store now.

You can count on Scooter for the latest innovations in electronic hardware like power protected control centers, a complete variety of cable assemblies and other unique hardware items for your computing and electronic needs. And, of course, you'll find those "common" electronic parts you've been mail ordering... and waiting for...on your dealer's Scooter rack.

Finally, you can find the quality, the selection and the latest developments in electronic components and accessories in one place: under the big green Scooter sign in computer stores across the country. So, when you're going for the best, go for the green. Scooter High Grade Electronic Components.

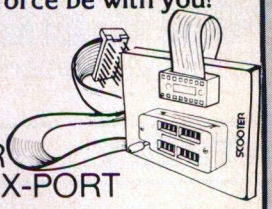
Go for the green!

If your favorite computer dealer doesn't have Scooter components yet, send this coupon to:
ohm/electronics
746 Vermont Ave.
Palatine IL 60067
(312) 359-5500

SCOOTER™
High Grade Electronic Components

Let the Ø-force be with you!

#ZFXP-3
$20.00

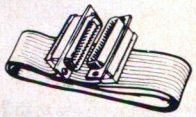


SCOOTER Ø-FORCE X-PORT

Scooter's zero-force external game port with 3-foot cable gives you easy game port access outside your computer. Avoid damage to fragile pins and internal computer port.

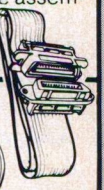
Get the connection!

#DB25-P-4-S
$21.75



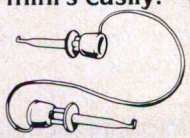
D-SUBMINIATURE CABLE ASSEMBLY

With 25-pin male to female connectors and 4-ft. cable. Compatible with Paper Tiger and other printers using D-submini connectors. Also: 36-pin male to female Centronics and Epson type cable assemblies with
4-ft. cable (#CCAP4S): **$29.80**
6-ft. cable (#CCAP6S): **$32.95**



Test sub-mini's easily!

#20412W
$3.69

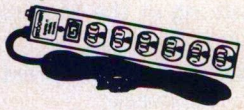


MINI-HOOK TEST LEADS

Easy access method of solderless connection and testing for delicate subminiature parts. Light weight and Finger-eze Hypo Action permit direct hook up to delicate wires.

Centralize and protect your power!

#SSP-6
$47.95



SCOOTER GUARD-IT CONTROL CENTER

Scooter's Guard-it six-outlet strip offers surge protection plus one-switch power control of all computer components. Rated at 15 amp, 125 volts. Maximum spike current: 4000 amps. Clamping spike voltage: 350. Power indicator light. Push-to-reset protected circuit breaker. 6-foot grounded cord. UL listed.

N3-8

My dealer _____ doesn't
(DEALER NAME)
carry Scooter yet. Please let me know where I can buy Scooter products now.

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP

Circle 52 on Reader Service Card

THE EDITOR BYTES BACK

the editor

WE'E off to greet springtime and the West Coast Computer Faire with a special 128 page issue, our largest to date. Lest we forget, the dates for the **big** show of the year is March 18-20 at **Brooks Hall,** in San Francisco's civic center. Hope to see you there, and at the **Anaheim Applefest** in April.

From time to time in **Call -A.P.P.L.E.,** we often feature assembly language programs which use Applesoft's powerful ampersand **(&)** command, which has the capability of passing parameters directly from an Applesoft program to a machine language subroutine. **John Crossley** of **Apple Computer, Inc.** really set the Applesoft world on fire with an article in the first issue of the **Apple Orchard** on the use of Applesoft internals, a corrected version of which appeared in **Call -A.P.P.L.E. In Depth 1.**

This now-famous article is no doubt singly responsible for the huge influx of ampersand-related programs that have appeared in this and other magazines over the last two to three years. **Craig Peterson,** in **Call -A.P.P.L.E. In Depth 1,** introduced the concept of combining several ampersand utilities in one program, enabled with a command parser which distinguished the several parsed ampersand commands and routed them to the appropriate subroutine.

During 1982, a number of commercial programs appeared on the market embodying this premise. Somewhere along the line, we managed to fire off five of them to staff writer **David Sparks,** who has pulled together a comprehensive review with diagrams, 8x10 [glossy?] charts and everything. His conclusion? They're all good. Turn to page 37 and see if you agree.

As we get firmly in gear for 1983, we are seeing more and more, and lower priced 16k RAM cards making the Apple scene, the end to a means, so to speak, of getting the most from your 64k of memory. **Exploring the World of Apple** welcomes **Coleman Logan** to our pages with a good deal of information on how your Language Card or 16k RAM card is turned on and off and how to use it to best advantage. His **Switchcraft: Memory Management** is one of several articles on this subject currently in the works

Another newcomer is **Robert Plamondon,** who in **Anatomy of an Eamon Adventure** explains artfully how this well known game series works, and additionally provides a program listing of the **Base 2** program. At the same time, be sure you don't miss our two Eamon specials which are described in this issue's **Bytes From A.P.P.L.E.**

Also appearing for the first time, **Charles Wells** presents an in depth review of **Transforth,** one of a multitude of variants of the popular **Forth** programming language. Forth is a "stack-oriented" language which is ideal for applications where you want to add a command to those already in the language, and has the advantage of running much faster than BASIC, almost like a compiled language. Wells' review will fill you in on the details.

Recently staff writer **Ralph Swerdlow** and his wife took hiatus from their daily routine and toured Europe. At our suggestion, they stopped in Rotterdam to visit another of our staff writers and his family. You can read about it in **A Visit with Cornelis Bongers.**

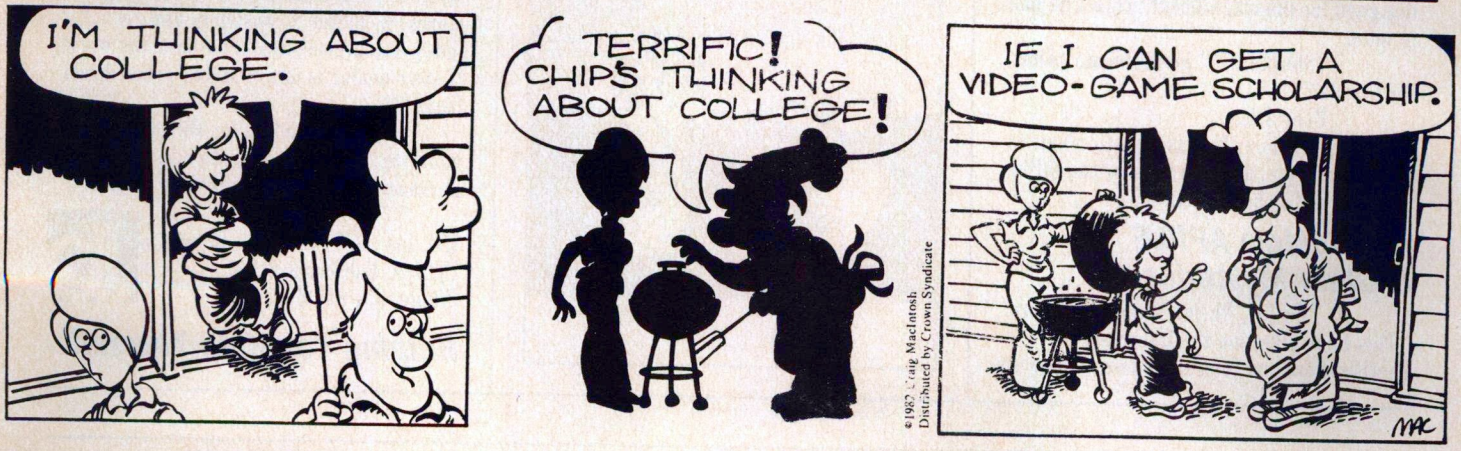
Two stories relating to CP/M appear in this issue. Our **CP/MUFFIN,** ably abetted by **David Sparks,** is a utility to convert CP/M files to DOS, while **Call -A.P.P.L.E.** engineering consultant **Murray Arnow** found and fixed a CP/M bug which shows up only on the Apple //e. The cure is his program **FIX APCPM**

Pascal users are not forgotten in this issue with **Terry Reilly's Jeanne's Procedures: A Guide to the Pascal Editor** and **Lawrence O'Glasser's Seeing Double Safely.**

Continued on page 45

BOOT

Craig MacIntosh



New Book

IT'S JUST GREAT

№ 1 IN THE SERIES

IJG

THE CUSTOM APPLE & Other Mysteries.

CREATE A WHOLE NEW WORLD OF POSSIBILITIES FOR YOU AND YOUR APPLE.

Dateline: California, January 1983.

J.R. Programmer, bit diddler and hardware hacker extraordinaire', wished that his APPLE computer had 6522 I/O capabilities available.

To have that one enhancement would open new vistas in data acquisition and display, complete his project, and make him millions.

"Gosh, wouldn't it be nifty?" exclaimed J.R.

But alas, the lack of appropriate hardware prevented him from accomplishing his goal.

J.R. looked everywhere in his local computer store for an interface or expansion board to solve his dilemma, but none were designed for the magnificent project he had in mind.

"Shucks . . ." he stormed, "what am I to do?"

Poor J.R.

But then, just as he was about to give up all hope, a splash of color caught his eye over in the book section . . .

The Custom APPLE & Other Mysteries.

"Oh joy, Oh joy!" cried our hero. "This book gives me the information, specifications and references I need to do the hardware enhancements, whether I am a beginner or an expert, and provides me with the basic information required to write the software, common to many of the projects, myself. This book is just crammed with all kinds of neat stuff!"

J.R. left the store, also an IJG book dealer, a very happy man.

A Hardware and Software Modification Guide.

The Custom APPLE & Other Mysteries provided J.R. with a number of data acquisition and control projects with camera ready printed circuit layouts like the 6522 application interface board, an 8-Bit D/A and A/D converter, a sound and noise generator board, an EPROM burner board, an APPLE Slot Repeater, and included information on the APPLE as a square wave generator, the control of two stepper motors, connecting two 6502 systems, and lots lots more.

Plug-In To Power And Get Turned On!

The Custom APPLE & Other Mysteries is available for $24.95 at computer stores, B. Dalton Booksellers and independent book dealers around the world. If your dealer is out of stock, order direct from IJG.

Include $4.00 for shipping and handling. Foreign residents add $11.00 plus purchase price. U.S. funds only please.

IJG, Inc. 1953 West 11th Street
Upland, California 91786
Phone: 714/946-5805

IJG

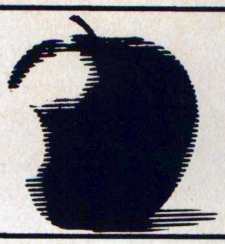
Helping You Help Yourself.

© IJG, Inc. 1982

Circle 51 on Reader Service Card

TM APPLE and APPLE II Trademarks of APPLE Computer Inc.

BYTES FROM A.P.P.L.E.



Val J. Golding

JUMP is a 6502 assembly language instruction, nearly identical to the BASIC GOTO. It is also a term we use in-house when a story is continued from one section of *Call -A.P.P.L.E.* to another. In the instant case, we went overboard talking about our big March issue a page back, so we are going to have to be brief and to the point here.

In *The Editor Bytes Back,* we told about the feature story *Anatomy of an Eamon Adventure* and mentioned an accompanying special (actually two). If you are into adventure games and have not yet run across **Don Brown's** *Eamon Adventure* series, you've been missing out, and we certainly don't want you to miss out on the opportunity to gleam onto the whole collection in one swell foop.

Eamon Adventure diskettes are included in our *As Is Software* series of public domain diskettes, available to members at $4 each. We currently have 32 of the Eamon series, which if you bought them separately would total some $128. Now here's the kicker: we'll add a 33rd disk, containing the base program listed in this issue, and let you "shoot the works" for just $95, so here's a great opportunity to acquire a superb group of games and also save $37!!! This offer will expire April 30, 1983 and must be phone ordered or postmarked by that date. Let's call this special **All About Eamon,** because we can use that to tie in to another pitch in this same column.

For those of you who have already ordered some of the Eamon disks, we haven't forgotten you. We have set up a second special — let's call it the **Eamon Special** — and you can select any ten *Eamon Adventures* from the total 32 for just $35. This special also ends on April 30th.

And now the tie-in we mentioned. **All About Dos** is the title of the third issue of *Call -A.P.P.L.E. In Depth,* and as we write this, is within a week or so of going on the press. Right now we are looking at about 250 pages, printed on heavy duty stock, of handy utilities, technical explorations, DOS speedups, tutorials, and reference material. If we're still on schedule, there will be a full page ad in this issue. Look for it and pick up on a whole new world of DOS information.

500% Faster Loading of Textfiles, Integer, Applesoft & Binary and Now Six Powerful Extra Commands in Standard DOS 3.3

30 out of 60 bought it on the spot when DAVID-DOS™ was demonstrated at the Original Apple Corps, UCLA.

Automatically supports an Integer or Applesoft Rom Card in any slot (without configuring).

Six New Commands

1. TLOAD Speed loads all Text Files.
2. TLIST Lists all Text Files to screen/printer.
3. DUMP Binary/Ascii to screen or printer.
4. DISA Disassembles Binary to screen/printer.
5. AL Prints program Address & Length.
6. HIDOS DOS moves itself to Language Card

Like Other DOS Commands

These six commands operate identical to existing DOS commands. Use A or A$ for address and L or L$ for length. Enter them on the Keyboard and use them in Basic programs with the familiar D$.

100 Sectors in 7 Seconds

TLOAD loads a Visacalc Template of a hundred sectors in 7 seconds and TLIST scrolls it to screen or printer. TLOAD speed loads Sequential and Random Text Files starting at $1000 or as specified by A. It loads the entire file or a smaller length specified by L.

List Any Textfile

TLIST lists Sequential and Random Text Files to screen and printer.

DUMP screen dumps memory with Ascii along the right side. Use A and L for specific dumps. Without them it starts at $00 and stops with return.

Disassemble Machine Code

DISA disassembles machine code to screen. Use A and L for specific routines.

Variable Speed Scrolling

TLIST, DUMP and DISA Features:

1. Print with PR#
2. Key operated variable speed control.
3. Instant pause with Space Bar.
4. Exit anytime with Return Key.

10K More of Memory

HIDOS command checks for a language card in slot O and moves DOS residing in memory to the card at $DD00. Himem is set at $BE00 adding 10K of free memory for programs. The Integer Card is still automatically supported in any slot. $300 to $3D0 is kept free.

Apple II and Applesoft are trademarks of Apple Computer inc. Visicalc is a trademark of Visicorp.

Two More New Commands

HIDOS has all previous commands and features operational and adds a 7th command FIND, Find searches 64K memory in 3 seconds for any Hex sequence up to 31 bytes long and prints the addresses to screen. Hidos has an optional 8th command DATE configured for Mountain Hardware clocks and is useable in basic programs.

Use with Most Software

DAVID-DOS is totally accurate and speed loads without over-writing. Great care was taken to keep DOS standard and compatible with most software. All DOS entry addresses have been preserved. Only two routines were moved to allow more room in command tables. DOS is the same length. Init areas were used for DAVID-DOS. Two extra variables are used in Zero Page.

Install in Seconds

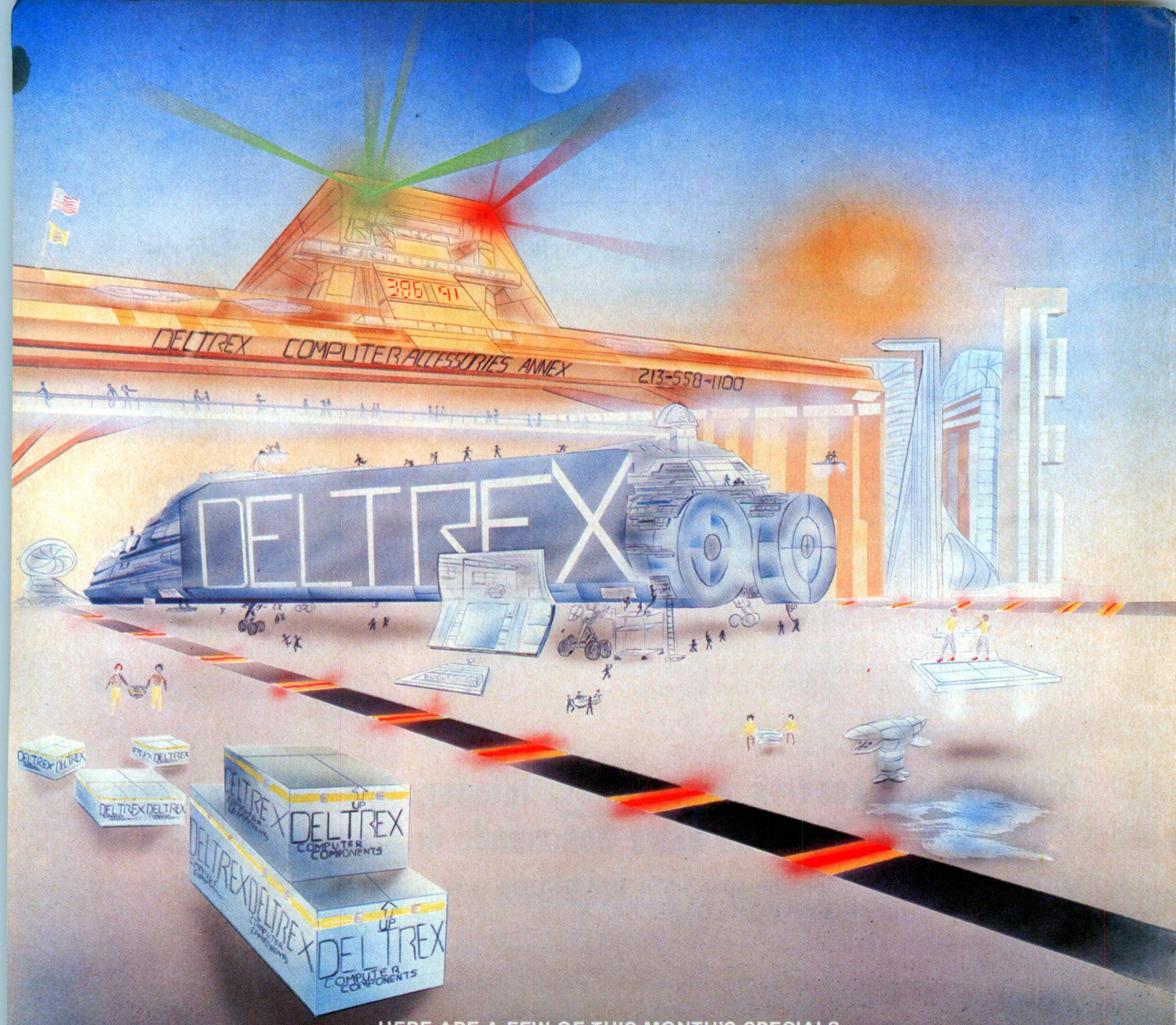
Catalog Free Space is also provided. DAVID-DOS installs in seconds. Requires a 48K Apple II with DOS 3.3. Comes with instructions.

California residents add 6½% sales tax. **$39.95** Foreign add $2.00.

DAVID DATA™

To Order: Send Check To
12021 WILSHIRE BLVD., SUITE 212A
LOS ANGELES, CA 90025 (213) 478-7865

Circle 61 on Reader Service Card



HERE ARE A FEW OF THIS MONTH'S SPECIALS

	Manufactures List Price	DELTREX Price		Manufactures List price	DELTREX Price
DELTREX LOWER CASE chip	$ 50.00	$ 9.95	BASF 5'' DISKS 10/box	$ 40.00	$ 24.98
M&R UNIVERSAL MODULATOR	69.95	49.95	T G APPLE joystick	59.95	34.99
KENSINGTON SYSTEM SAVER fan	89.95	64.95	KRAFT APPLE joystick	64.95	38.99
PROMETHEUS VERSACARD	199.00	139.00	VIDEX ENHANCER II	149.00	99.98
WIZARD 80 COLUMN CARD	249.00	147.00	RANA ELETE 1 drive	379.00	269.95
VIEWMAX 80 COLUMN CARD	219.95	149.95	MICROBUFFER II 16K	259.00	198.95
VIDEOTERM 80 COLUMN CARD	345.00	227.95	VISICALC 3.3	250.00	179.95

IF YOU ARE PAYING LIST PRICE, THEN YOU ARE PAYING UP TO 70% TOO MUCH.

DELTREX™ 2265 Westwood Blvd. Dept. #575 Los Angeles, CA 90064 (213)558-1100

Orders $1,000 and below add 3% shipping. ($3.00 min.) CA add 6% (L.A. County 6½%). COD $2.00 extra

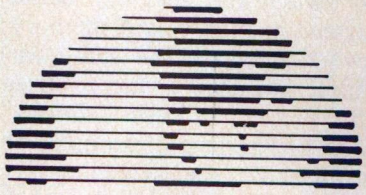
SHOP THROUGH OUR ENORMOUS SELECTION USING OUR 24 HOUR MODEM. (213)202-6886

Circle 63 on Reader Service Card

EXPLORING THE WORLD OF APPLE:

Switchcraft: Memory Management

Coleman Logan



THE primary purpose of this article is to show how you can utilize the 16K of RAM memory that is available to you on a 16K RAM card located in slot 0. A secondary purpose is to show how a language ROM card (Applesoft or Integer) can be integrated into your system. I have used the Microsoft 16K RAM card, and its documentation, to develop the material contained in this article. The operation of most 16K cards is similar to the Microsoft card.

To get an orientation of the location of the available memory, first examine the figure. This figure relates the different address spaces with the physical location of memory banks. A memory bank is simply a contiguous section of memory; and in the case we are examining, there are four memory banks, each with 12K bytes whose address space starts at $D000 and ends at $FFFF.

The first memory bank, (chips 1–6) in the figure, represents the main board ROM. The next two memory banks are like a two-headed snake, the only physically independent section is at the top. The first of these two memory banks consists of chips 7 through 12, and the second consists of chips 13, 14, and 9 through 12. Although each of these memory banks contains 12K of RAM, the two banks only add up to 16K of physical memory. (So that's why they call it a "16K RAM card"!) Also, just to keep you on your toes, the vendors of these cards refer to the first of these memory banks as "Bank 2" and the other as "Bank 1." To keep things straight, I suggest that you keep one eye on the figure and try not to think numerically.

The right-most memory bank in the figure is shown for the benefit of those who are lucky enough to own a BASIC ROM card (either Applesoft or Integer), in addition to the 16K card. This memory bank contains 12K of ROM, and is the counterpart of the 12K of ROM on the main board. (For a complete discussion of a ROM card in a non-zero slot, see *Call -A.P.P.L.E.*, May 1982, p. 69).

(The diagram of the RAM card in the figure is drawn for the purpose of easily relating physical memory with the address space that the programmer uses. The diagram should not be used to attempt to locate a bad chip on a 16K card.)

Since all four of these memory banks have the same address space, the question of memory management is obvious. How do you get the 6502 to read or write the addresses corresponding to the physical address space that you want it to read or write?

Before answering this question, there is another attribute of memory management that is rather mind-boggling when first encountered. The 6502 can have a read-address space that is physically different than its write-address space! This means that LDA $F800 followed by STA $F800 could copy a byte from the main board into the "same" location on the 16K board. Example 3 below illustrates this attribute of memory management.

The answer to the question of memory management is summarized in Table 1. How the information in the chart can be used will be demonstrated through a series of examples.

Before getting to the examples, let's make a correspondence with the chip figure and an Apple][+ and an Apple][, after the system has been booted up using the 3.3 Master Diskette:

Apple][+: Chips 1–5 contain Applesoft;
Chip 6 contains Autostart or Old Monitor ROM;
Chip 7 contains Programmer's Aid #1;
Chip 8 is not in use;
Chips 9–11 contain the Integer Package — Basic, the Mini-Assembler, Sweet 16, and Floating Point Routines;
Chip 12 contains a copy of the Autostart ROM; and
Chips 13–14 are not in use.

Apple][: Chip 1 might have Programmer's Aid #1;
The socket for Chip 2 is usually empty;
Chips 3–5 contain the Integer Package;
Chip 6 contains Autostart or Old Monitor ROM;
Chips 7–11 contain Applesoft;
Chip 12 contains a copy of the Autostart ROM; and
Chips 13–14 are not in use.

Unmodified DOS switches between the main board memory bank and "Bank 2" of the 16K RAM card. This bank switching is included in the DOS commands, FP and INT. These commands not only switch memory banks, they also reset Applesoft and Integer program pointers, an action not always desired. The first example below shows how you can move from the Applesoft memory bank to the Integer memory bank (the one containing the mini-assembler) and back to the Applesoft memory bank without losing the Applesoft program pointers.

SWITCHCRAFT

The following series of examples are specific instances of using the switch codes summarized in Table 1 to switch the memory banks illustrated in Figure 1. A memory bank switch operates the same as other Apple switches — merely accessing the proper address "throws" the switch. The first 3 examples demonstrate how the switching is done directly from the keyboard in the monitor mode, and the remaining examples show switching under program control. Only the last two examples assume that slot 4 is being used.

Example 1.

Let's assume you are developing an Applesoft program and you want to enter a short assembly program using the mini-assembler, and when done with that, return to the Applesoft program. (If you let DOS do the memory switching with a]INT, the Applesoft program pointers would be lost.)

First, we'll assume you have an Apple][+ (if you don't, hang in there, we'll get to you in a minute), and have booted with the Master Diskette, placing the Integer Package into the RAM card. The following steps would be keyed in:

1)]CALL -151	To get into monitor
2) *C080	Switch to Bank 2 Read Mode
3) *F666G	Will produce mini-assembler prompt, !

When done with the mini-assembler, to return to Applesoft:

1) !$FF69G	To return to monitor
2) *C082	To return to main board address space
3) *3D0G	Will produce Applesoft prompt,]

Now for those of you who have the Apple][and are developing an Applesoft program using Applesoft on the RAM card — to go to the mini-assembler and return to your Applesoft program, the following steps would do the trick:

]CALL -151	To get into monitor
*C082	Exit the RAM card
*F666G	Will produce the mini-assembler prompt, !

When done with the mini-assembler, to return to Applesoft:

!$FF69G	To return to monitor
*C080	To return to RAM card address space, (Chips 7 through 12)
*3D0G	Will produce Applesoft prompt,]

Example 2.

Assuming an Apple II+, place a copy of the main board monitor residing in $F800-$FFFF (Chip 6) to the same address space in slot 0 (Chip 12):

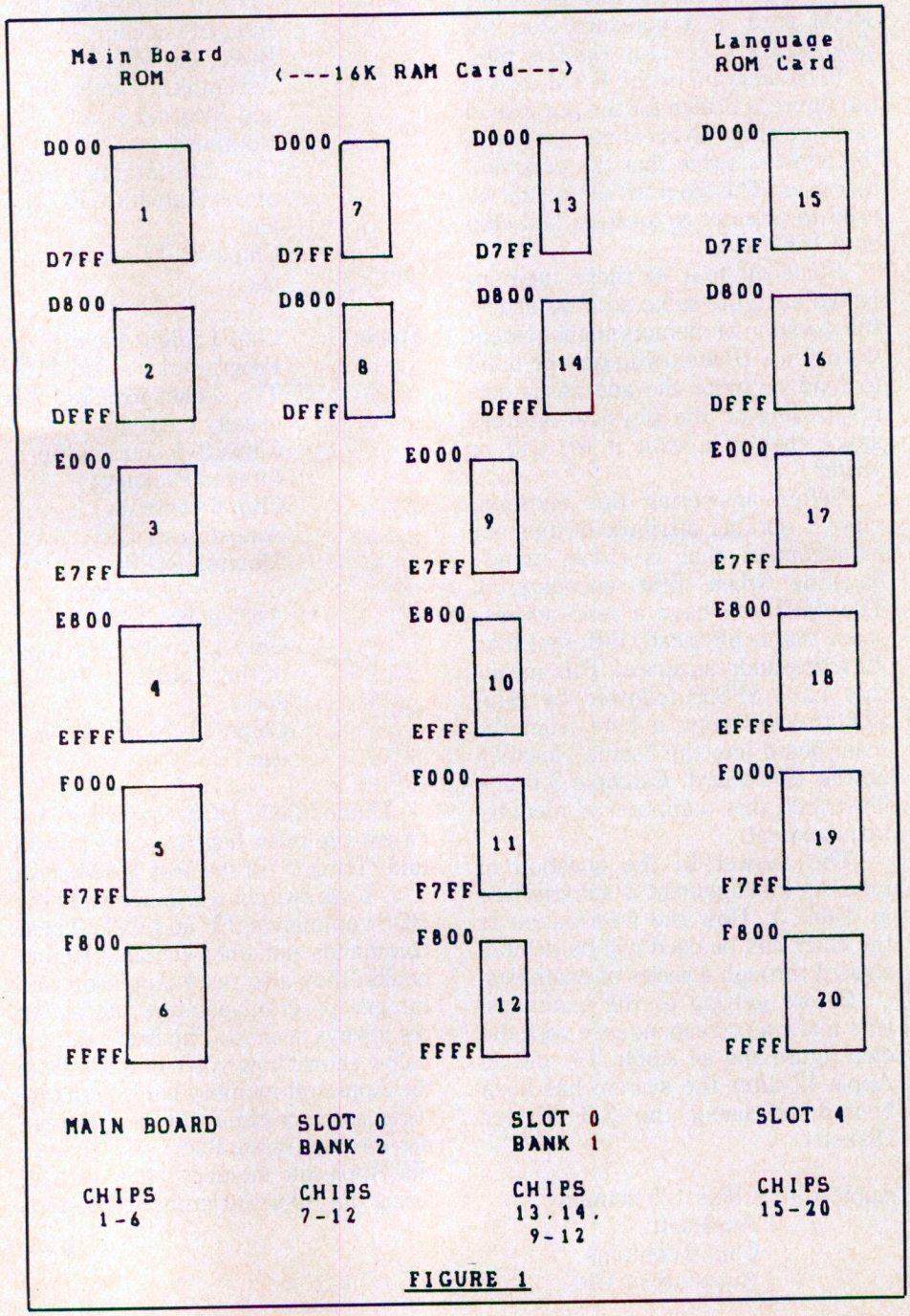


FIGURE 1

1)]BSAVE MONITOR,A$F800, L$800
2)]CALL -151
3) *C089 C089 Set Bank 2 in Write-only Mode
4) *3D0G
5)]BLOAD MONITOR Read main board addresses, write Bank 2 addresses

The normal booting process leaves the 16K card in write-only mode. If no memory switching has been done after a boot, only the BSAVE and BLOAD commands are necessary for this example.

Example 3.

Without using DOS, move a copy of the main board monitor residing in $F800-$FFFF (Chip 6) to the same address space in slot 0 (Chip 12):

1)]CALL -151
2) *C089 C089 Set Bank 1 in Write-only Mode, with read addresses on Main Board
3) **F800<F800.FFFFM Read main board addresses, write bank 1 addresses

Example 3 is an especially important utility for the general use of the 16K card. In case you use the 16K card without booting with the Master Diskette, be sure to place a copy of the monitor onto the 16K card before setting the RAM card to read/write mode. If you don't do this, and then transfer control to the monitor while in this read/write mode, you will find yourself in the very embarrassing position of having to turn the power off to regain control of your Apple! But in this danger lurks an opportunity — your RAM copy of the monitor can be modified to your heart's content.

One more word of caution to those who want to use the RAM card without booting with the Master Diskette — the normal return, 3D0G, will not work if a memory bank containing BASIC is not active in the read mode.

Example 4.

Using the mini-assembler place a machine language program into the Bank 1 address space starting at $D100 (Chip 13). The program will beep the speaker twice, using monitor routines located in Slot 0 (Chip 12).

For this example, let's assume that we have an Apple][+, and we are in Integer Basic mode (the active address space are Chips 9-12) — the normal situation for Integer on an Apple][+. (If you have an Apple][and wish to run this example, first boot with a regular disk, then insert 3.3 Master diskette and BLOAD INTBASIC,A$D000. This action configures the 16K card as if you had an Apple][+ and had booted with the 3.3 Master Diskette.)

First, enter the following from the keyboard:

```
CALL -151          To move to monitor
*C08B C08B         Set Bank 2 into Read-write mode
*F666G             Call mini-assembler (in Chip 11)
!D100:JSR $FBDD    Beep
! LDA #$FF
! JSR $FCA8        Delay
! JSR $FBDD        Beep
! RTS
!$FF69G            Exit mini-assembler
*C080              Place Bank 2 into Read mode
*3D0G              Go back to Integer Basic
```

Now the machine language program is in place.

How can this program be called from Integer Basic executing in Bank 2? The switching of memory banks should normally occur in the lower 48K of memory. Since the machine language program will be called from BASIC, the switching program can be located at $0300.

```
0300 - LDA $C088   Place Bank 1 in Read mode
0303 - JSR $D100   Call the machine language program
0306 - LDA $C080   Place Bank 2 in Read mode
0309 - RTS         Return to Basic program
```

Table I Reference table for switches

Slot 0, Bank 2

C080	Turn on Read Flag (Read-only Mode) (Read Chips 7-12, "Write" Chips 1-6)
C081	Turn off Read Flag (Cease Reading Chips 7-12)
C081 C081	Turn on Write Flag (Write-only Mode) (Read Chips 1-6, Write chips 7-12)
C082	Turn off Read and Write Flags (Read/"Write" Chips 1-6)
C083 C083	Turn on Read and Write Flags (Read/Write Mode) (Read/Write Chips 7-12)

Slot 0, Bank 1 ("Folded" Bank)

C088	Turn on Read Flag (Read-only Mode) (Read Chips 13,14,9-12; "Write" Chips 1-6)
C089	Turn off Read Flag (Cease Reading Chips 13,14,9-12)
C089 C089	Turn on Write Flag (Write-only Mode) (Read Chips 1-6, Write Chips 13,14,9-12)
C08A	Turn off Read and Write Flags (Read/"Write" Chips 1-6)
C08B C08B	Turn on Read and Write Flags (Read/Write Mode) (Read/Write Chips 13,14,9-12)

Slot 4 (Applesoft or Integer BASIC ROM card. Reduce addresses C0C0 and C0C1 by $20 to use Slot 2)
(If Slot 0 is active, enter C082 before enabling slot 4)

C0C0	Turn on Read Flag (Enable) (Read Chips 15-20)
C0C1	Turn off Read Flag (Disable) (Read Chips 1-6)

(To exit slot 4 memory, disable Slot 4 with C0C1)

Then the double beep program can be executed from BASIC by CALL 768 (Remember 768 is $0300).

This example demonstrates how the BASIC programmer can use the entire 4K bytes of unused memory on the 16K card (Chips 13 and 14) for a series of routines located there. The 10-byte routine at $0300 can be used to execute all of the routines if the starting address of each routine first overloads locations $304 and $305, and then 768 is CALLed.

Example 5.

Assume the double beep program of example 4 has been entered into Bank 1. Assume that an Applesoft ROM board is in Slot 4. (To get DOS to recognize this fact:

```
*A5B8:C0
*A5C0:C1
```

as indicated in *Call -A.P.P.L.E.*, May 1982, p. 69)

Assuming that Applesoft is active (Chips 15 through 20 are in the read mode), how do you get the double beep program to execute?

Again, just CALL 768, but with the following program located at $0300:

```
0300 - LDA $C0C1    Exit Slot 4
0303 - LDA $C088    Place Bank 2
                    in Read mode
0306 - JSR $D100    Call double
                    beep program
0309 - LDA $C08A    Disable Slot 0
030C - LDA $C0C0    Enable Slot 4
030F - RTS          Return to Slot
                    4 space, the
                    Applesoft call-
                    ing program.
```

Example 6.

Assume the same situation as example 5, but assume an Integer ROM board is in Slot 4. Everything will be done exactly as Example 5, except that the CALL 768 will be an Integer CALL 768, rather than an Applesoft CALL 768.

ASSEMBLER/TEACHER

This program makes your APPLE* teach you assembler! We assume you know BASIC already, and now you'd like to learn APPLE* assembler. You need three things (two heavies and a friend) . . .

1. The APPLE II Reference Manual (heavy),
2. A good 6502 assembler manual (another heavy: ask for both at your APPLE* dealer), and
3. ASSEMBLER TEACHER, a friendly program that takes you right through the learning barrier!

ASSEMBLER TEACHER will soon have you eating those manuals for breakfast. And you'll never outgrow ASSEMBLER TEACHER, since it has tools an expert can use even more than a novice. Here's how ASSEMBLER TEACHER works:

- ASSEMBLER TEACHER prowls your APPLE*s memory page by page, drawing a map of what it finds.
- ASSEMBLER TEACHER shows your CPU running in slow motion on your screen, so you can see exactly what the operations are doing to the registers and the stack (a great debugging tool).
- ASSEMBLER TEACHER exposes the mysteries of hexadecimal, two's complement, character and other data representations, instantly translating from any version to all others.
- ASSEMBLER TEACHER gives you the concepts you need to cope with assembler, using simple on-line lessons that grab you by the keyboard and make you play your way to knowledge.
- ASSEMBLER TEACHER is written by John Fairfield, a Ph.D. in Computer Science who regularly teaches assembler. A RAM Mini-Assembler is included.

"Rating: A . . . a great help to anyone interested in learning machine language programming . . . and the price is right." Peelings II (Jan. '82)

Ask for ASSEMBLER TEACHER at your dealer, or order from:
ASSEMBLER TEACHER
Computer Works, Inc.
P.O. Box 1111, Harrisonburg, VA 22801
(703) 434-1120

$44.95 MC VISA DOS 3.3

*Registered trademark of Apple Computer, Inc.

Circle 65 on Reader Service Card

Improve Your Programs on the APPLE II

Your favorite routines belong with the most efficient "power-boost" possible for the II/II+. **THE ASSEMBLER DEVELOPMENT SYSTEM** provides the cleanest path to interface the powerful 6809 with DOS.

STELLATION TWO

The Lobero Building P.O. Box 2342
Santa Barbara, Ca. 93120
(805) 966-1140 Telex 658439

Circle 64 on Reader Service Card

A Call -A.P.P.L.E. REVIEW:

Transforth

Charles Wells/Reviewer

TRANSFORTH is a language for the Apple][and the Apple *III* written by Paul Lutus, who wrote Apple Writer. It is marketed by *Insoft Incorporated,* 10175 Barbur Blvd, Suite 202B, Portland, OR 97219. Both versions cost $125. In this article I shall describe in some detail its features and contrast it with two other languages I am familiar with — Pascal and BASIC.

Immediate and deferred mode

A computer language system must translate instructions written in the language into a string of numbers which cause the computer to do what the program writer wants. (Actually, the translated instructions tell the computer to do what the author *wrote*, which unfortunately is not always what he *meant).* If the system translates each instruction as soon as it is written and carries it out right then, it is said to be in *immediate mode.* When you work in immediate mode, you are working with the computer in much the same way as you would work with a pocket calculator.

Another possibility is that you can store the instructions you have written to be executed after you finish writing them, or alternatively to be saved to tape or disk. This is called *deferred mode.*

Transforth, like BASIC but unlike Pascal, can be used in both immediate and deferred modes. The way you use immediate mode is the same in both languages. You type a sequence of instructions in the language, press return, and the computer carries the instructions out. In both cases you have available the standard Apple editing functions to change or correct what you have written before you press return. The ability to add new instructions to Transforth, which BASIC does not have, means you can custom-tailor the language for use in immediate mode, which I have found to be a tremendous advantage. More about that later.

To use BASIC in deferred mode, you give the line you type a number. The BASIC interpreter (the program which implements BASIC instructions) stores the line. After you have written your program you can either RUN it or save it on disk.

In Transforth or Pascal, you call up an editor and use it to write your program. To write a Transforth program, you can use the editor supplied with the language or one of several commercially available editors. The one supplied with Transforth is quite small and simple; it has the Apple editing commands plus a very few others - like automatic line numbering. Its small size means it can live in the memory at the same time as the language and the program you are writing (until the text and compiled program together grow to more than about 24000 bytes) — it is what is called a "coresident editor." This is convenient because for debugging you can go back and forth instantly between the editor and immediate-mode Transforth. That is what I usually do for small programs.

For large programs, I have enjoyed using Apple PIE to write programs. This cannot be coresident, so it takes a little longer to switch back and forth between editor and language, but PIE has a great many powerful commands which the Transforth editor lacks, such as global search and replace and the ability to enter lower case.

The instructions supplied with Transforth say that any editor program which produces ordinary DOS text files may be used to write Transforth programs. They also give a short program (easily added to the language as an instruction) for using Applewriter files to write programs. I have not tested this aspect of the language.

Compiled vs. interpreted languages

When you write a program in BASIC and save it to disk instead of running it, what you save is a compressed version of the program you wrote. Words such as PRINT and GOTO are encoded in one-byte codes to save space. This encoded form does *not* constitute the translation into machine code necessary to implement the instructions. That translation is done by the interpreter on the spot, when the program is run, and the resulting machine code is carried out immediately.

Pascal and Transforth do not work like that. When you write a program in either language, the whole program is translated into machine language *before* it is run. The process of translating into machine language is called "compiling" the program and Pascal and Transforth are therefore *compiled* languages. (Actually, some forms of Pascal on microcomputers, such as UCSD Pascal, are compiled into an intermediate code called "p-code" which is interpreted by an interpreter when it is run. The interpreter is much smaller than the BASIC interpreter because the p-code is very much like machine language already. For this reason and because you have to compile a whole procedure before you can run it, UCSD Pascal is a compiled language in spirit, if not in literal fact.)

Even when used in immediate mode, a line of Transforth code is compiled before it is executed. This takes place so fast as to be unnoticeable.

In deferred mode in both Transforth and Pascal, you normally save both the written text of the program as a text file (in case you need to modify it later) and also the compiled version to be run later. Since the code saved to memory is already compiled, a Pascal or Transforth program

—●→

generally runs faster than a similar BASIC program, because the BASIC interpreter has to translate each instruction and find the memory location of every variable every time it is executed. More about speed below.

There is a difference between Transforth and Pascal, however: A Pascal program is saved to disk by name like a BASIC program, and called by name to run it. A compiled Transforth program is also given a name, but the name is *added to the language as a new instruction* and the modified language is saved to disk. The instructions in Transforth are called words, and the name of the program is just another word. (I'll give some examples below.) When you have loaded the language into memory, typing the word will run that program. A Transforth "program" thus contains the entire language and compiler. This is not as much of a handicap as it seems, since the language as supplied by *Insoft* complete with compiler takes up only about 8000 bytes (less if you omit some supplied instructions you don't use). In contrast, the Applesoft interpreter takes up more than 10,000 bytes.

One reason Transforth can be smaller and compile so fast is that it has a much simpler syntax than BASIC or Pascal. This needs a little digression to explain.

Parsing

Consider what happens when a BASIC statement like:

PRINT X + Y*Z

is executed. If you think about it, you will realize that the *last* thing that can be done is print the answer, even though the word PRINT comes first in the statement. The quantity X + Y*Z has to be calculated first. And to do that, Y*Z has to be calculated before it can be added to what is in location X. And before Y*Z is calculated you have to find where the numbers denoted Y and Z are stored and make them available to the machine-language routine which multiplies numbers. And so on.

The process of rearranging the instructions into the order they are to be executed is part of what is called *parsing.* Parsing may be described roughly as the process of finding the inherent abstract structure of a sentence — what is its subject, verb, and so on. (Recognizing the words of the language is also part of parsing, but I won't say anything about that here.) When the BASIC interpreter parses a line, one of the main effects is to arrange the tasks in the order in which they are to be performed. After that, each task can be translated into machine instructions. BASIC parses an instruction every time it is carried out, whereas Pascal parses it once, as part of the compilation process, and stores the parsed and translated instruction as part of the compiled program.

Transforth adopts a different strategy: With a few exceptions, it expects you to write the instructions *in the order in which they are to be carried out.* This is what you have to do when using Hewlett-Packard and Novus calculators; it is called "Reverse Polish notation".

That's not a Polish joke; it refers to the eminent Polish logician Jan Lukasiewicz who discovered in the 1920's that writing function symbols before the things they operated on (for example + AB instead of A + B) meant that you could do without parentheses. His system was widely called "Polish notation" by logicians. From our point of view he was writing things down in the order you do them from right to left. Writing them left to right is more natural (at least to people who speak languages which are written left to right) but because Polish notation already used the other order, the left-to-right order has come to be called "reverse Polish" or "RPN". Transforth is an RPN language. (It uses parentheses to enclose comments in a program — it has no other use for them!)

Other languages of the Forth family also reverse Polish notation. They differ from Transforth in other ways, for example in being interpreted instead of compiled.

As a first approximation, the BASIC command

PRINT X + Y*Z

(which is written in almost the same way in Pascal) in RPN would be

Y Z * X + PRINT.

That does not take into account the fact that X, Y and Z are memory locations, not numbers, and you cannot add and multiply memory locations. You have to FETCH the numbers X, Y and Z before you can do anything to them. In Transforth, the symbol for FETCH is @ and the symbol for PRINT is . (period). (These symbols look funny but they can be learned fairly quickly and save a lot of typing.) Thus in Transforth, PRINT X + Y*Z is

Y @ Z @ * X @ +.

What this sequence of instructions actually does is this: The symbol Y is the name of a variable. That means somewhere previously in the program there is the line

VARIABLE Y

This "variable declaration" sets aside five consecutive bytes in memory to hold a five-byte floating point number (the same sort of numbers that Applesoft Basic uses). Transforth also allows one and two-byte numbers to be manipulated, about which I shall discuss more later.

Unlike Pascal, variable declarations can occur anywhere in the program prior to their first use. Unlike BASIC, variables *do* have to be declared.

When Y occurs later in a program, the address of the location referred to by Y is put on the "stack," which is a list of the numbers currently being worked with (just like the stack on a reverse-Polish calculator). The last datum put on the stack is said to be on the "bottom" of the stack. (This caused me no end of confusion when I first read the instructions, since in much of the computer science literature the last thing put on the stack is on the *top.)*

The word "@" replaces the number on the bottom of the stack by the number stored beginning in that memory location. Thus after executing "Y @" the stack has the number in the location denoted by Y on the bottom of the stack. You can put a number directly on the stack by simply typing the number. You put a number into a memory location by

typing the number, the location, and "!," which means "store" but which I usually pronounce "eek!." Thus,

```
365 Y !
```

puts the number 365 into variable Y. You can get the effect of the BASIC POKE by simple typing the explicit memory location:

```
365 16384 !
```

puts the number 365 into the five bytes starting at 16384.

The word "!" expects to find two numbers on the stack. If it finds only one, you get an error message.

The word "*" takes the last two numbers on the stack and multiplies them. Thus typing

```
5 7 *
```

leaves 35 on the stack. Similarly, "+" adds the last two numbers on the stack and "." prints the last number on the stack (removing it in the process).

I found the reverse Polish notation used by Transforth very easy to get used to because I have used a Hewlett-Packard calculator for several years. Since it follows the natural flow of calculation it allows you to write a program by thinking of what you would do next and then writing it. Some people are horrified at the idea of reverse Polish. All I can say is that different people find different things easy or hard. Some people can write a BASIC program using 30 different variables all with two-letter names and remember what each one means. I cannot. If find the restriction to two-letter variables in Applesoft Basic sufficiently restrictive as to destroy the usefulness of the language for me. As I say, different people find different things hard.

However, I do suspect that many people do not give reverse Polish a fair chance. One reason may be that many of the published programs in RPN languages keep some of the variables on the stack instead of storing them in a named location and calling them up as needed. It is hard to read such a program. It is also hard to write one. The usual reason variables are kept on the stack is to save time. I have found that Transforth (in contrast to pocket calculators) is sufficiently fast for most purposes that it is not worth the effort to keep things on the stack.

As an example, here are three versions of simple computation used in solving a differential equation. The first version is in BASIC, the second is a direct translation into Transforth using the same variable names, and the third is a version in Transforth which keeps everything on the stack. Putting everything on the stack makes it run about 20% faster. That might occasionally be worth it when the procedure is to be run thousands of times, but usually it is not worth the effort in programming and the resulting unreadability.

BASIC Version

```
10  H = .005:X = 0:Y = 1
20  DEF FN F(Y) = X + Y * Y / (X
    + 9)
30  FOR K = 1 TO 1000:Y = Y +
    H * (FN F (Y)):X = X + H:
    NEXT
40  PRINT Y
```

Transforth version with named variables

```
    0 VARIABLE X
    1 VARIABLE Y
.0005 VARIABLE H
 :    F Y @ DUP *
      X @ 9 + /
      X @ + ;
 :    EULER
      F
      H @ *
      Y @ + Y !
      X @ H @ + X ! ;
 :    RUNIT 1000 0 DO
      EULER LOOP Y@.;
```

Transforth version with values kept on stack

```
 :    F DUP * OVER 9 + / + ;
 :    EULER
      OVER OVER F
      4 PICK * +
3     SWAP 3 PICK + SWAP ;
( EULER2 EXPECTS H, X AND Y
TO BE ON THE STACK IN THAT
ORDER )
( IT LEAVES H, X+H AND THE
NEW VALUE OF Y ON THE STACK )
 :    RUNIT .0005 0 1 1000 0 DO
      EULER LOOP.;
```

OVER is a Transforth word which copies the next-to-bottom entry in the stack to the bottom. (If the language were consistent with the instruction manual OVER would be UNDER). DUP duplicates the bottom value. N PICK copies the nth from the bottom to the bottom, and SWAP switches the bottom two. Transforth has several other words for manipulating the stack, including rolling it up and down.

A colon-semicolon pair defines a new instruction whose name is the first word after the colon. Thus in programs 2 and 3 the word F is defined and then used in the word EULER (named after Leonhard Euler, who discovered this method of solving differential equations).

To sum up, Transforth uses reverse Polish notation, so that the compiler has to do very little in the way of parsing. As a result it is smaller and faster than a Pascal compiler. This allows interactive programming in the same way as BASIC but with a faster, compiled program as the end product.

But wait — you can interactively produce a fast, compiled program in BASIC, too, using one of the BASIC compilers that are available for many microcomputers. Why use Transforth? There are several reasons. Transforth, unlike BASIC but like Pascal, is *modular* and has much better *control structures.* Let's look at each of these properties in turn.

Modularity

A language is modular if it is designed in such a way that a large program can be written in independent parts (modules) which are invoked by calling them by name. Each module is designed to do a specified task (it pays off put a bit of effort into making sure you have specified exactly what a module will do) and is tested separately to see that it fits the specifications before it is incorporated into the program. Most modern languages which have been designed since Fortran and BASIC have included features designed to make them modular. It is hard to do modular programming in most dialects of BASIC because you cannot call subprograms by name and because you have to be careful to avoid using the same variable name or line number twice in different parts of the program.

The ability to define new words, like EULER in program 2 and 3, make Transforth as modular as any language. Every new defined word has the same status as the words already in the language and can be used anywhere they make sense in a program. Thus a Transforth program is built up by defining words which do pieces of the action you want; these words are used in words which put them together into a bigger piece, and so on, until the last word you define is your whole program. Each word can be debugged before it is used. Moreover, once these words are compiled that become a permanent part of the language. (Rest

assured that you can *FORGET* words you no longer want). In immediate mode calculations, it is as if you could insert a new button into your calculator doing any calculation you might want to do more than once.

Control Structures

Applesoft Basic has three control structures. You can choose which of two branches of a program to take using IF...THEN or ON...GOTO, and you can loop using FOR...NEXT. More modern languages such as Pascal and the newer Basics available on other computers have additional control structures.

For example, in many languages, IF...THEN statements can have an ELSE clause, allowing you to specify what to do if the condition following IF is false. This can be imitated in Applesoft by using a GOTO, but that makes the program hard to follow. Pascal and other languages have a WHILE...REPEAT structure of one sort or another, causing a part of the program to be repeated as long as a certain condition is true. Transforth has both of these structures, as well as certain variations on them which I will not get into here.

To someone used to Pascal, the Transforth controls look strange because of the RPN nature of the language. For example, the IF...THEN controls have this syntax:

```
<word> ... <word> IF <word>
<word> ... <word> THEN
<word> <word> ...
```

The words between IF and THEN are executed only if the bottom of the stack is nonzero when IF is executed. Thus 2 2 + IF BLAH THEN GLUG would execute BLAH, then GLUG whereas 2 2 - IF BLAH THEN GLUG would execute only GLUG. The trouble is, that corresponds with the English sentence "If 2+2 is nonzero then do BLAH go on and do GLUG, otherwise just do GLUG." The word IF in Transforth comes *after* the test condition instead of before, (so it means "If the bottom of the stack is not zero then do..." and the word THEN comes after the conditional part, not before. I think such statements would have been less confusing to newcomers if Transforth used ENDIF instead of THEN. However, I admit that I am now completely used to the construction so perhaps the choice of words is no big deal.

Transforth has a WHILE construction not in BASIC which makes a Transforth program much easier to construct and read, again taking into account the reverse Polish nature of the language. Its syntax is

```
BEGIN <word> ... <word> WHILE
<word> ... <word> REPEAT
```

Here, if the words between BEGIN and WHILE leave a nonzero number on the stack then the words between WHILE and REPEAT are executed, after which the words between BEGIN and WHILE are executed again. This continues until a zero is left on the stack when WHILE is reached. If that happens, execution continues with the first word after REPEAT. Again, WHILE comes after the condition it test instead of before, as in Pascal and English.

Transforth also has constructions with ELSE, LOOP, UNTIL and CASE which give it as much in the way of control structures as most modern languages, but which suffer from the same problem of being initially confusing because the test word comes after the condition tested instead of before. I believe most people can get used to this arrangement fairly quickly.

These control structures eliminate the need for GOTO statements entirely. Transforth does not have GOTO, and the amazing thing to me is that I have never wanted to use it. There have been occasions when I have been programming in Pascal where I did use GOTO (which is available in many versions of that language). Since Pascal also has fairly powerful control structures, I can't understand why I want to use GOTO in Pascal but not in Transforth. Perhaps it has something to do with thinking in reverse Polish fashion, but I honestly don't know.

Examples

As an illustration of modularity and the use of WHILE, Program (4) is a preliminary version of a program which enables me to record the grades of my classes in a disk file. I have since elaborated it so that it has a menu, alphabetizes the list, computes averages and so on. This early version is a stage the program went through in its development.

```
PROGRAM (4) GRADEBOOK PRO-
GRAM
50  200 1 ARRAY CLASS

VARIABLE LAST
:   ADDNAME
    PAD LAST @ CLASS
    MOVELN
    LAST @ 1 + LAST ! ;

:   GETNAME
    PRINT " NAME: "
    PAD READLN
    BEGIN PAD LENGTH WHILE
    ADDNAME GETNAME
    REPEAT ;

:   GRADEINPUT
    CLASS SWAP 20 + +
    B! ;
    ( 90 3 17 GRADEINPUT GIVES
    THE ( 17TH STUDENT A
    GRADE OF 90 ON THE
    ( THIRD TEST )

:   GRADEOUTPUT
    CLASS SWAP 20 + + B@ . ;

( 3 17 GRADEOUTPUT PRINTS
THE GRADE
    ( OF THE 17TH STUDENT ON
    THE THIRD ( TEST )
```

The line 50 200 1 ARRAY CLASS declares an array CLASS each of whose entries is 50 bytes long, with 200 entries. The 1 means it is one dimensional. You put the address of the 17th entry in CLASS on the stack by typing 17 CLASS. A 20 by 20 two-dimensional array NUMS of 5-byte entries would be declared by the line 5 20 20 2 ARRAY NUMS and the entry in NUMS(5,10) would be obtained by writing 10 5 NUMS @ (note the reversal!).

The variable LAST keeps track of the number of entries made so far. The first 20 characters will be used for the name of the student; the last 30 to record grades of as many as 30 different tests.

The word ADDNAME is designed to put a name typed at the keyboard into the array. It is used in the next word GETNAME which prompts for the name and then inserts it into the array. Notice that after GETNAME inserts the name into the array, it calls itself again. All this is under the control of a BEGIN...WHILE... REPEAT statement. PAD is the name of a special Transforth input

buffer; the word PAD puts the address of the buffer on the stack. LENGTH is a Transforth word which gives the length of the string starting at the location on the bottom of the stack. (The user has control over which character determines the end of a string.) Transforth has several other string manipulating strings, including one very handy one which tells which of two words comes first in alphabetical order.

Thus the instruction PAD LENGTH will put a nonzero number on the stack unless the user types return without typing a name. Therefore the user will continually be prompted for a new name each time he enters one until he types return in response to the prompt.

The fact that GETNAME can call itself, is an illustration of the fact that Transforth is "recursive".

GRADEINPUT and GRADEOUTPUT are used to store and retrieve grades for 30 tests in the last 30 bytes of the array CLASS. A grade has to be a number between 0 and 255 — that is all that can fit into a byte. The words "B@" and "B!" moves numbers in and out of one-byte locations in the same way that "@" and "!" move them in and out of five-byte floating point locations. There are also words for handling two-byte numbers which are therefore like numbers in Integer Basic, Tiny Pascal and some other languages. (They are even stored high byte first for ease in interfacing with machine language subroutines.)

The array CLASS is therefore very much like a record in Pascal. The first twenty bytes are for a name, the last thirty for small integers. Instead of declaring a "type" of variable with space for 20 letters and 30 small integers, I simply implement this arrangement by hand. If I needed another array with the same characteristics I would have to implement it by hand again. In Pascal, I could define a record type with those characteristics and then create as many variables of that type as I want. In this way Transforth is not as powerful as Pascal.

As I understand it, the <BUILDS DOES> system in standard Forth could be used to accomplish this, too. Transforth does not have <BUILD DOES>. It is difficult for me to know how much of a loss this is. The published examples I have seen of the use of <BUILDS DOES> in Forth mostly seem easily implementable in Transforth in other ways. One exception is mentioned below.

Error Checking

There is one major thing Transforth lacks which both BASIC and Pascal have. That is error-checking. The instruction

```
3.1416 5 10 NUMS !
```

(referring to the array NUMS described previously) puts the number 3.1416 in NUMS(10,5). No problem with that. But if I typed

```
3.1416 10 22 NUMS!
```

I would not get an error message even though I exceeded the bounds of the array NUMS. What is worse, the number 3.1416 would be stored in the memory location where NUMS(22,10) *would* be if it existed — in this case, about 250 bytes beyond the end of the space reserved for NUMS. Thus it could easily wipe out some other important part of the program. (You have no idea how many times I have wiped out the editor doing things like that.)

Transforth does almost no error checking at all, except for a few things like having too few or too many numbers on the stack, IF without THEN, and the like. This is deliberate and, I understand, consistent with other languages of the Forth family. It enables the compiler and the program to be small and fast.

You could easily put error checking into the word NUMS, however. Define an array

```
5  20  20  2  ARRAY  NU
```

and then the words

```
: BOUNDCHECK DUP DUP 0 >
  SWAP 21 < AND 3 PICK DUP 0
  > SWAP 21 < AND AND ;
: PRINTERR PRINT " ARRAY
  INDEX OUT OF BOUNDS " ;
: NUMS BOUNDCHECK IF NU
  ELSE PRINTERR THEN ;
```

AND puts 1 on the stack if the bottom two numbers are both nonzero, 0 otherwise. It removes the two numbers it tests. Thus BOUNDCHECK puts 1 on the stack if the bottom two numbers are both between 0 and 20 (it leaves them on the stack).

Then NUMS will put the address of the appropriate item in NU on the stack if the indices are within the appropriate bounds, but will print an error message if not. In Transforth, as in BASIC, array indices actually start at 0, but the error-checking above as written won't let you call, for example, 0 5 NUMS.

Inclusion of error-checking of that sort might be useful, although I have made only occasional use of it myself. After the program is debugged, the declaration of NUMS could be rewritten to get rid of the error-checking so that the program would run faster. That's not hard, but that sort of thing can be done even more easily using the <BUILDS... DOES> in some other languages of the Forth type such as the standardized version FIG-Forth. (You can introduce error-checking into a whole class of words and then get rid of it in the whole class if you wish.)

That may be one place where FIG-Forth is definitely superior to Transforth — but I remain skeptical. When I first started using Transforth, I ran into a lot of mysterious hangups, program crashes and the like, but I have adapted and now feel that the necessity of paying attention to what each word is going to do and (most important) what it is going to change rather than depending on the compiler to take care of me is a good discipline. It is particularly liberating after using Pascal, whose insistence that only variables of the right type be used in a given situation I have always found confining.

However, I think I would have found Transforth impossible if I had not already had a certain amount of programming experience. (I am not a professional programmer, which I modestly contend makes me a better reviewer.) My conclusion is that *Transforth is not for the beginning programmer.* Indeed, for learning programming, I don't know of anything that beats Pascal.

Documentation

There is another reason neophytes should avoid Transforth. The instruction manual is generally accurate but woefully incomplete. The first version of it did not even tell you about some of the features of the language (such as how to give an instruction to Apple's Disk Operating system from the editor). It was also very short on examples. *Insoft* has issued an update which includes some more information and a few more examples, but many, many more examples are

—●→

needed, and lots of discussion of the peculiarities of the Transforth words as well.

On the other hand, the *Insoft* people are very good indeed about answering your questions by letter or phone, and they say they are working on a tutorial manual which should be a great improvement.

Speed

A Transforth program usually runs faster than a BASIC program, anywhere from a little bit faster (if the BASIC program is very small) to two or three times as fast. For example, Program (2) iterated 1000 times runs in 18 seconds, compares to 20 seconds for the BASIC program (1). Program (3) run 1000 times takes 15 seconds. On the other hand, I wrote a Transforth version of Jim Galbreath's benchmark program in the September, 1981 *Byte* which ran in 108 seconds compared to 265 seconds for Applesoft.

Generally the bigger the Applesoft program, the slower it will be compared to Transforth — that is the advantage of a compiled language. On the other hand, my fellow Northeast Ohio Apple Corps member Bill Cowan compiled the Applesoft version of Galbreath's benchmark program using *On-Line's Expediter* and it ran in 52 seconds. That may be because of Expediter's use of two-byte integer arithmetic which Transforth does not have. Although Transforth can save you lots of *space* by storing integers in one or two bytes, it converts them all to five-byte floating point for arithmetic, and so using small integers saves you no *time*. (Applesoft itself has the same defect.) This might matter in some applications, particularly in real-time games.

The literature supplied with Transforth says that it is slower than FIG-Forth because the latter keeps two-byte numbers on the stack whereas Transforth keeps five-byte numbers on the stack. Actually, there are floating-point extensions of FIG-Forth available now; I would speculate that when operating with floating point those extended languages are not any faster than Transforth.

Perhaps *Insoft* ought to consider producing a machine language module providing integer arithmetic operating off a separate stack for those applications where that would matter.

Miscellany

Transforth has all the mathematical functions that Applesoft Basic has, such as SIN, COS, TAN, ATN, EXP, LOG, SQRT and RND. It also has MOD (modulo), FRAC (fractional part) and INT (integer part). It will print numbers in four different formats: dollar, fixed decimal, and both scientific and engineering floating point. It also has a command to print a byte in hexadecimal notation.

Transforth has a number of words allowing flexible input-output routing. You can specify disk, memory or device for input or output. There are words for moving numbers, strings, bytes or whole files from input to output. This notation is independent of the Apple DOS terminology and makes one wonder if *Insoft* is planning to implement Transforth on other computers.

There are words making it quite easy to define new words in machine language, although presumably the machine code has to be relocatable if it is to be included in the word definition. It is also possible to set aside space in memory by absolute location and to CALL a machine language program located there. (Setting aside memory has other uses, too, such as avoiding the part of memory Apple sets aside for high-resolution graphics.)

Transforth also has a word NOTE which sounds a note on Apple's speaker with pitch and duration determined by the top two numbers on the stack.

Unlike any Applesoft compiler I ever heard of, the basic Transforth language as supplied operates entirely independently of the Applesoft interpreter. A compiled Transforth program will run on any Apple and Transforth can make use of the memory space normally occupied by the Applesoft interpreter if you have a language card. (It does use monitor routines, however.) The exception is a module supplied by *Insoft* for high-resolution graphics, which does use Applesoft interpreter routines. This module is a simple but powerful implementation of turtle graphics; with Transforth's capability of defining new words, you can write a program to draw a shape, give it a name, and behold, that shape is part of your language.

Transforth uses ordinary Apple DOS files. You can save uncompiled programs as text files and call them up by name. Enormous programs can be compiled by having one program file call the next one. A fellow Apple Corps member who uses FIG-Forth has told me that the most irritating thing about FIG-Forth is its use of a different disk operating system which is rather rigid and hard to use in editing programs. In all fairness, though, Fig-Forth programs run on a variety of different machines whereas Transforth is available only on the Apple.

A Transforth program can be saved in "locked" mode. In that mode it cannot be accessed as a language and you can't get out of the program into the monitor for the purpose of looking at its code. (At least, I haven't figured out how to.) If you sell a program written in Transforth, the *Insoft* people want you to see it in locked mode and to notify them that you are doing it. They do not charge for royalties.

Summaries

In my opinion, Transforth is an excellent language for someone who has some familiarity with programming and needs a general purpose language with more power and speed than BASIC. Its interactive nature makes it easier (though less safe) to use than Pascal, but it has about the same power and structure as Pascal.

Another reviewer of Transforth said that it had some of the features of each of Applesoft, Pascal and FIG-Forth but not all of any of them. For that reviewer, that was a complaint. In my opinion, Transforth strikes the *right* compromise for many purposes and should be considered by anyone looking for a relatively inexpensive but powerful and fast language for the Apple.

☐ VISISCHEDULE ☐ SUPERCALC ☐ VISICALC ☐ WORDSTAR ☐ D.B. MASTER ☐ MULTI PLAN ☐ VISIFILE ☐ dBASE II ☐

☐ AZTEC ☐ SUBMARINE COMMANDER ☐ WIZARDRY—SCENARIO I ☐ PREPPIE ☐ CANYON CLIMBER ☐ DEADLINE ☐ STAR RAIDERS ☐ CENTIPEDE ☐ ZORK II

☐ LETTER PERFECT ☐ DATA PERFECT ☐ FILE MANAGER 80 + ☐ SCREENWRITER PROFESSIONAL ☐ PFS: GRAPH ☐ THE SENSIBLE SPELLER ☐ VISITREND/PLOT

NEVER INVEST IN SOFTWARE AGAIN!

unless you can "test" it first.

United Computer's SOFTWARE RENTAL LIBRARY
. . . the nation's largest, now RENTS The Most Popular Software Available
at **15%** of manufacturers' retail price*

You could spend hundreds of dollars for software that may be "unfriendly' or inadequate for your specific applications. United Computer's SOFTWARE RENTAL program eliminates the guess work, and insures that the choice you make is the right one for you.

If you decide to buy after you have tested 100% OF THE RENTAL CHARGE WILL APPLY TOWARDS PURCHASE. All rentals are for 7 DAYS. Simply return software within 7 days, and rent another. . . and another. . . and another, until you find the right software for your business or entertainment applications.

There are now 2 different plans to choose from:

Join the **Game Group** for only $50.00 per year and receive your first computer game rental **FREE.** Then rent as many games as you like for only 15% of Mfrs. Sugg. Retail Price.* Minimum order, 3 game rentals	Join the **Business Group** for only $125.00 per year and receive your first rental **FREE.** Then rent as many business application programs as you like for only 15% of Mfrs. Sugg. Retail Price.*

REMEMBER, THESE ARE NOT DEMOS, BUT ORIGINAL UNRESTRICTED SOFTWARE PROGRAMS
(complete with manuals in original manufacturers' packages)

To Immediately Order, or for more information:

Money Orders or credit cards

MasterCard VISA

Checks allow 2 weeks

Toll Free CALL 1-800 992-7777
In California CALL 1-800 992-8888
In L.A. County CALL 1-213 823-4400

*plus postage and handling. Some programs may require 4-6 weeks delivery.

☐ FROGGER ☐ CHOPLIFTER ☐ GORF ☐ DAVID'S MIDNIGHT MAGIC ☐ EASTERN FRONT (1941) ☐ ZORK I ☐

FUTRA COMPANY

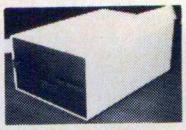
(213) 328-8951 (800) 421-5006
TWX 910 349 6211
AGENFTRA TRNC

Mail Order: **Department C**
P.O. Box 4380
Torrance, CA 90510-4380

Retail: **OMC** **20695 S. Western Ave.**
Suite 124
Torrance, CA 90501
(213) 328-1760

apple computer
Sales and Service

NEW FUTRA FLOPPY ADD-ON



$259.00

The new Futra 5¼" Floppy Disk Drive is fully software transparent with DOS 3.3 and 3.2.1 operating systems, as well as Pascal and CP/M. Add it to your present drive and controller for greater capacity and faster access. Just plug it in and go to work.
The Futra Apple-beige unit is built around a highly reliable Siemens system with over 10,000 lifetime hours. A shielded connecting cable is attached.
A Futra drive makes an excellent investment in preventing down time. At this price you don't have to be down a week waiting for your Apple drive to be fixed. Keep a spare Futra Drive just in case.
The Futra Drive features On-Track Head Seek with a precision lead screw to position the head onto the correct track. Retries and disk-to-disk copy errors are virtually eliminated.
Futra Drive has a faster Track Access Time at 25 MS than Disk II (40 MS), and a faster Average Access Time at 298 MS (Disk II at 463 MS).
The unit has a one-year warranty and MTBF (Mean Time Between Failures) is 8500 power-on hours. That's almost 3 years based on 8 hours per day.

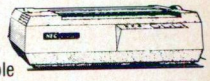
Manufactured by Quentin Research.

PRINTERS

OKIDATA
- Microline 82
- Microline 84

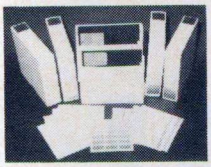
C. ITOH
- Prowriter
- Starwriter
- Printmaster

NEC 8023A



Grapplers Available

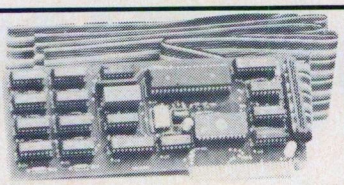
FUTRA Floppy Diskettes



$17.50
box of 10

Plain Wrap!
Why pay for fancy packaging? It's what's inside that counts. Try our single-sided, double density, soft sectored diskettes. They're so good . . . they're awesome!

MICROTEK Apple Dumpling-64
Spooler/Buffer for Apple Computers



Dumpling 64-K Buffer
$225.00

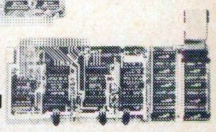
In addition to the standard graphics features, the Apple Dumpling-64 is the next logical extension to the graphic dump boards. Now you can get all the standard graphics features and 64K of buffer memory. Why tie up your computer waiting for your printer to catch up? Just dump the information into the buffer area and away you go. You're not just buying an interface, you're buying time.

The Apple Dumpling-64 allows full use of hi-resolution functions:
Rotation, Inversion, Double Size, Emphasis Print, Dual Page Dump, Chart Recorder Mode and control over all variables for text and graphics.
Of course, there's PASCAL AND CP/M compatibility, ideal default conditions, and a standard 2 year warranty. Available for most graphics printers.

MICROSOFT

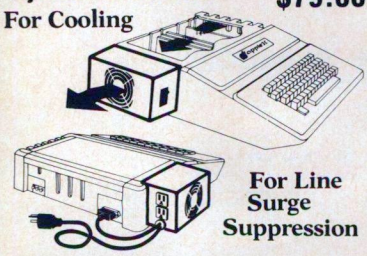
$229.00
Z-80 Soft Card with CP/M

$89.00
16K Ram Card



Futra brings you the industry standard at it's lowest price ever. When you can get the trusted Microsoft quality, why buy anything else?

KENSINGTON MICROWARE
System Saver Fan **$79.00**
For Cooling



For Line Surge Suppression

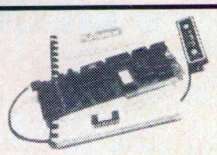
LPS II/PENTRAK SYSTEM from Gibson Laboratories
Light Pen System for Apple II

$259

The LPS II is the only true High Resolution Light Pen System with full software support for the Apple II Computer. High Resolution pictures, diagrams and other graphics can be easily drawn on the screen of the Apple II. PENTRAK, the light pen driver, lets you easily create your own Applesoft programs for Light pen graphics. PENPAINTER is a color sketching system which allows the user to create an infinite variety of patterns to be used for filling-in the sketched areas. Area refilling allows various combinations of patterns to be tried and changed. Hi-Res Text generation is a standard feature of the PENTRAK driver, allowing simultaneous use of multiple user-defined character sets. List price $349.00.

Possible Applications:
- Computer Aided Design
- Animation & Game Play
- Process Control
- Interactive Education
- Computer Aided Drafting
- Logic Design Simulation
- Business Graphics
- Text Editing
- Graphic Arts
- Menu selection
- Circuit Analysis

VIDEX INC.
VIDEOTERM

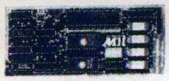


$239

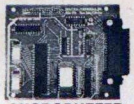
When you're looking for a 80 column video board, why settle for anything less than Videx? Futra Co. makes it possible for you to own the best.

Manual Switch Plate	$19
Electronic Switch Plate	32.50
Micromodem II Firmware ROM	25.00

PRACTICAL PERIPHERALS
Buffer Interfaces



MICROBUFFER II INTERFACE

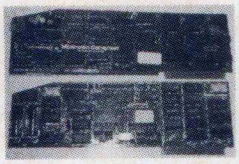


MICROBUFFER MBP-16K

Free your computer system from the binds of your printer!! Now you can get a buffer built into the interface that fits into the computer or a buffer that you install in the printer.

• Microbuffer II 16K	235.00
• Microbuffer II 32K	279.00
• MBS-8K Epson Serial (Printer)	149.00
• MBS-16K Epson Parallel (Printer)	135.00

MULTIFUNCTION CARDS



Mt. Computer CPS Multi-function card	165.00
Prometheus Versa Card	159.00
Mt. Computer Apple Clock	125.00

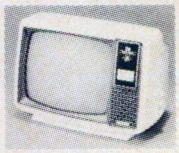
HAYES MICROMODEM II $269

The Micromodem II is a complete communication system for the Apple II

NOVATION APPLE CAT II $269

Buy the Apple Cat II, the modem that grows with your needs. Find out about Novation's add-on features

MONITORS



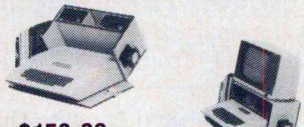
BMC 12" Green
$95.00

NEC 1212 Color Composite
$349.00
Amdek Color I+
$319.00

APPLE SOFTWARE — GAMES

Automated Simulations		Time Zone	89.96
Temple of Apshai	29.00	Frogger	31.45
Crush, Crumble & Chomp	25.00	Soft Porn Adventure	23.96
Broderbund Software		**Sirius Software, Inc.**	
Apple Panic	23.96	Gamma Gobblins	26.95
Serpentine	31.50	Sneakers	23.96
Star Blazer	24.00	Escape from Rungistan	25.50
Midnight Magic	29.70	Kabul Spy	31.45
Arcade Machine	53.95	Bandits	27.96
Choplifter	32.50	Phantom Five	26.95
Budgeco		**Sir Tech**	
Raster Blaster	23.96	Wizardry	37.50
Infocom		Knight of Diamonds	27.96
Zork I	32.50	**Strategic Simulations**	
Zork II	32.50	Computer Quarterback	27.00
Deadline	44.95	Warp Factor	27.00
Muse Software		Cartels and Cutthroats	27.00
Three Mile Island	35.95	President Elect	27.00
Castle Wolfenstein	23.96	Computer Baseball	27.00
On-Line Systems		Tigers in the Snow	27.00
Jawbreaker	23.00	**Sub Logic**	
Threshold	34.50	Flight Simulator	26.80

COOL STACK — SENTRY II



$159.00

Protect your investment! In today's world of "Smash and Grab" it only takes a burglar a few seconds to make off with a computer that took you months or years to save for. Not only does the Cool Stack-Sentry II lock down, but it cools and organizes your system. Be safe not sorry.

ACRYLIC PLASTIC PRINTER STANDS

$24.50*

Put your printer on a tilted pedestal. Now you can check your printer with just a glance, and save space by storing your paper under the printer. Specify Epson MX-80, NEC 8023A, or Okidata Microline 82.

Epson MX-100 **$34.50***

*Available in Smoke Bronze

DATA SUPPLIES

Flip Sort
Plastic Storage case **$22.95**

Scotch Products: For reliable data recording use the best diskette for the money. With reinforced center!

Box of 10 - 5¼" storage carton	$25.00
Scotch Head Cleaning Kit	$24.00

MOUNTAIN COMPUTER

Apple Clock	125.00
ROM Plus	125.00
ROM Writer	141.00
CPS Multi-function Card	165.00
Music System	355.00
X/10 Control	160.00
Expansion Chassis	602.00

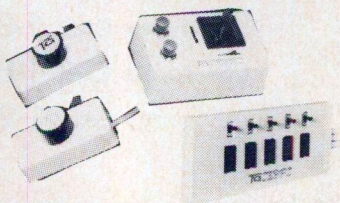
GENERAL APPLE SOFTWARE

Microsoft		**Ashton-Tate**	
Cobol 80	$562	DBase II	$549
Fortran 80	175		
ALDS	95	**Visicorp**	$CALL
TASC	149		
Micropro		**Continental Software**	
Wordstar	$359	Home Accountant	49.50
Datastar	202	Home Money Minder	29.70
Mail Merge	85		
Spellstar	129	**Stoneware**	
Calcstar	130	DB Master	$155
Supersort I	159	DB Master Utility Pack	89

Software Publising Corp.
Personal Filing Systems 87.50
Personal Report Systems 85.50

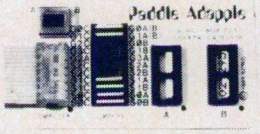
PFS: Graph 109.00

TG PRODUCTS



Joysticks	45.00
Game Paddles	27.50
Selectaport	45.00

PADDLE ADDAPPLE $27.00



Game I/O over loaded? Don't open your Apple. Get the two-plug Paddle-Addapple. Switch selectable and reasonably priced.

TERMS: Shipping: add 3% for product shipped within continental USA via ups surface (minimum $3.00). **If the order placed is prepaid with U.S. funds in the form of check or money order, a Total Charge of $3.00 for shipping is all you pay within the continental USA vis UPS surface.** Allow 14 working days for personal and company checks to clear. Credit card charges limited to $1000. No COD's. FPO, APO or orders outside continental USA call or write for shipping charges or add 20% to purchase price (any difference will be refunded). CA residents add 6% sales tax (LA County 6.5%). All offers subject to change or withdrawal without notice.

*CPM is a registered trademark of Digital Research.

Circle 12 on Reader Service Card

Anatomy of an Eamon Adventure

Robert Plamondon

How the EAMON Adventure System Works.

COMPUTERIZED adventure games have been surrounded by an aura of mystery that has nothing to do with their story lines. Simply stated, the mystery is this: no one knows how they work.

Like most mysteries, adventure programs are pretty straightforward once the hidden parts are revealed. To bring these secrets to light, let's look at the EAMON adventure system by Donald Brown, which is a set of BASIC programs for the APPLE][and available to members at $4.00 each on the A.P.P.L.E. "As Is" software series. EAMON is a good choice because it's a good system, it's all in BASIC, and it's in the public domain, which means that you can adapt or copy it at will. It does most of the things that all-text adventures do, so studying it can give you a good idea of how adventure games work.

The Structure of EAMON.

EAMON is a set of programs on disk which call each other as necessary, passing data through text files. The number of programs varies slightly depending on whether you have the original version or an altered one. In my own version I have merged several of the programs together, reducing the number of programs somewhat.

Warning: This gets a little intricate. If you don't care to know about the different programs, and just want to get the crucial stuff, skip this section.

The HELLO program runs when you boot the EAMON Master Disk. It loads an impressive picture of a dragon to the high-resolution graphics display to give you something to look at, then it runs a program called THE WONDERFUL WORLD OF EAMON.

THE WONDERFUL WORLD OF EAMON is the program that comes up with characters for you to play in the adventure. It prompts for your character's name, then tries to find an existing character of the same name in the file CHARACTERS. If it doesn't find him, it asks if you spelled if correctly (giving you a chance to try again). If you insist your spelling was correct, the program decides that you must be starting a new character and creates one for you, storing the data in the file CHARACTERS, and the name of your character and his record number to the file THE ADVENTURER. It also gives you a chance to read the instructions (Listing 1).

If the name you gave matches with an existing record in CHARACTERS, his name and record number are written to THE ADVENTURER. In both cases, the next thing the program does is to issue the command, "RUN MAIN HALL."

The MAIN HALL is where characters go to buy and sell weapons, buy armour and spells, and depart for adventures. The MAIN HALL reads the number stored in THE ADVENTURER, then reads in the character record of the same number from CHARACTERS.

After doing all the buying and selling that appeals to him, the player has two options: he can send his character on an adventure, or quit. If he quits, his updated character record is resaved in CHARACTERS. If he decides to go on an adventure, the character record is partially erased (although the data still exists in program variables), and the player is prompted to insert the disk with the adventure on it. The character's data is written to a file named FRESH MEAT, and the file EAMON.NAME is read. EAMON.NAME contains the name of the adventure program. The MAIN HALL then runs the program of that name, and the main adventure begins.

Okay, so it's complicated. The code that does all the file manipulations is a lot shorter than the explanations, though, and passing parameters between program is not the crucial part of EAMON, anyway. We'll get to the good stuff in a minute.

The reason why the character record is partially erased before the character embarks on an adventure is to make sure that if he dies on the adventure, he stays dead. As it is, there are only two ways to get your character's statistics back into the CHARACTERS file. One is to survive the adventure. The other is to use the RESURRECT program on the EAMON Utility Disk (which contains other interesting programs, as well). Resurrecting a character is considered cheating by purists, but everyone does it anyway.

The program called by the MAIN HALL usually prints out introductory material for the adventure; telling you what's going on, what your character is supposed to do, and so forth. This program in turn calls *another* program (don't worry, we're almost there): the BASE DUNGEON PROGRAM, which does the actual adventure. The rest of this article will deal with the BASE program.

A (Nearly) Infinite Loop.

The main activity of an adventure program is a loop: the program gets a command from the player, figures out what it means, goes to the appropriate routines to carry out the command, and returns to get another command. This loop continues until the adventure is completed, the character dies, or the power is turned off.

EAMON ADVENTURE

Getting commands is an easy job. All it takes is a statement like "INPUT A$." Breaking the command into words, and identifying the actions corresponding to the words is also pretty easy, it turns out. The routines that carry out the actions are usually fairly simple, too.

So where's the hard part? The individual parts are all easy, but there are a *lot* of individual parts. A program that puts them all together can be immense. This makes the design job difficult enough that most people who'd like to write adventures never get started. Fortunately, EAMON makes a good base for all sorts of adventures, and the design work for the programs has already been done.

One Program, Many Adventures.

One of the most useful things about the EAMON system is that the same BASE program is used (with minor alterations) with all the EAMON adventures. This is possible because all the information about a specific adventure is kept in text files, and this data is what controls the adventure.

For example, the information about which rooms connect where is kept in a text file, which is loaded into an array at the beginning of the adventure. This data controls where the adventurer can go. Monster data is likewise kept in files, so the creatures you encounter will vary from one adventure to the next.

The files for each EAMON adventure are:

1) EAMON.DESC, which contains the long descriptions for each room, artifact, special effect, and monster in the adventure.
2) EAMON.ARTIFACTS holds the data on artifacts (i.e., objects) in the adventure, including names, values, initial placement, etc.
3) EAMON.MONSTERS contains monster data, and is similar to EAMON.ARTIFACTS.
4) FRESH MEAT holds information on the character.
5) EAMON.ROOMS holds the information on which room connects where.
6) EAMON. ROOM NAMES holds the names of each room.

These files provide all the information required to run a unique adventure.

The Program Listing.

Reading the listing of the EAMON adventure system is necessary if you want to understand the way EAMON works. I have added a large number of remarks to the listing to make it more comprehensible (there are very few comments in the original program.) Once you understand how this program works, you should be able to write your own adventure program, if that appeals to you. If you want to write adventures, but don't want to design a new program, EAMON is the system for you. It's only available for the Apple][(as far as I know), but it would be fairly easy to convert it to run on other machines.

This is my version of the BASE program, which is a modification of John Nelson's modification to Donald Brown's original. John Nelson's version is the one on the Dungeon Designer Version 5.0; mine is Version 5.1, and previous versions are (as far as I can tell) Donald Brown's original creation. A number of routines have been changed. Some of these make the program run faster, some correct errors in coding or concept in the original, and some make the output more readable.

A Brief Overview of the Program.

Lines 100-900 make up the main loop. Lines 100-200 print out a description for the room and each monster and object in it. If something has been encountered before, its name is printed. If something is being seen for the first time, the long description is printed.

Lines 200–290 get the player's command and process it. The input is broken into a subject and verb (S$ and V$), and the verb is compared to the list of commands in the string array C$. If a match is not found, an error message is printed along with a list of acceptable verbs. If there is a match, the appropriate routine is jumped to in line 290. If the player just hits RETURN, rather than typing anything, his previous command is repeated.

Lines 300-900 control combat. Each monster in the room gets its chance to attack. Morale is checked for each monster: if the morale check fails, the monster flees from the room. A monster's morale decreases as it is wounded. Line 900 is "GOTO 100" — forming the (nearly) infinite loop.

Lines 1000–2000 handle initialization. Variable and arrays are set to their initial values, and data is read in from disk. The list of command words is in line 1920.

Lines 2000–3000 handle the end of the game. If your character lives, he gets to sell his treasures and return to the MAIN HALL program. If he died, the MAIN HALL program runs THE WONDERFUL WORLD OF EAMON, giving you the chance to start a new character.

Lines 3000–4000 handle movement. The program determines which direction you want to move, and checks to see if it's possible. If it is, you are moved into another room.

Lines 3600–3900 handle monster reaction. If the monster hasn't been met before, "dice" — random numbers — are rolled and compared to its "Friendliness." The monster will be either friendly, neutral, or hostile. If may follow you from room to room (it will always do so if friendly, never if neutral, and will sometimes chase you when hostile). This makes monster reaction important to the movement routine.

Lines 4000–5000 handle picking things up. It checks to see that the item is in the room, and isn't too heavy. If the character is unarmed, and the item is a weapon, the weapon becomes his "ready" weapon — the one he is prepared to fight with. The command "GET ALL" causes the character to pick up everything that's not too heavy for him.

Lines 5000–6000 make up the DROP routine, which is the GET routine in reverse. DROP is used to get rid of unwanted baggage, and is helpful in combat to arm a comrade who has lost or broken his weapon. If a weapon is on the floor, any unarmed combatant will pick it up. (GIVE doesn't work during combat, though perhaps it should.)

Lines 6000–7000 are the LOOK or EXAMINE command, which is usually used in the form of "EXAMINE SWORD." The long description in EAMON.DESC is printed if the thing you're looking at is in the room.

Lines 7000–8000 make up the combat routine. The combat system is suspiciously similar to that in *RuneQuest,* a popular fantasy role–playing game (not a computer game). Each creature has a chance to hit with each category of weapon (axe, bow, spear, and sword). These numbers are in the range of zero to something over 100. If a random number between 1 and 100 is less than the chance to hit, the creature hit its opponent.

Critical Hits do extra damage, and *Fumbles* cause the attacker to drop his weapon or hurt himself.

Armor absorbs damage, but reduces the wearer's chance to hit because of its weight and bulk. *Armour Expertise* is the skill of compensating for the hampering effects of wearing armour.

If a creature scores a hit, there is a chance that his skill at the weapon he is using will increase (practice makes perfect).

Lines 7700–7900 handle dying creatures. When a creature dies, its weapons and possessions fall to the floor, and his dead body (an artifact) is placed in the room. Its Courage is reduced to 75% of its current level in case it is resurrected by a POWER spell (having just died, it would think twice before risking its life again). If the player's character dies, the game is over (this is one way out of the nearly infinite loop).

Lines 8000–9000 handle the FLEE command. The character flees through a randomly selected exit. Sometimes your enemies follow, and sometimes they don't.

Lines 9000–10000 are the GIVE routine. You can give anything you are carrying to a monster. This sometimes makes them friendlier. If you use a number, as in "GIVE ORC 500" the program assumes you are giving that many gold pieces to the monster. Money makes a good bribe.

Lines 10000–11000 form the INVENTORY routine, which prints out the name of everything your character is carrying.

Lines 11000–12000 form the BLAST spell, which is used to wound opponents. All spells in EAMON decrease in effectiveness with use: if you have an 80% chance of success on the first attempt to use the spell, it is reduced to 40% on the second try, 20% on the third, and so on. This is not a very effective way of limiting spell use, but no one has changed it yet.

Lines 12000–13000 form the HEAL spell, which cures wounds on your character, but can't be used to heal his friends (someone ought to fix this).

Lines 13000–14000 are the POWER spell, which does unpredictable things — from causing things to vanish to raising the dead to collapsing the roof on your head. Many adventures have a BASE DUNGEON PROGRAM with a modified POWER spell, just for variety.

Lines 14000–15000 make up the SPEED spell, which increases agility and the base chance to hit. SPEED is nice because you can cast it on yourself before a battle (when spell–casting is safe), and it lasts a reasonably long time.

Lines 15000–16000 are the SMILE command, which is a way to try to make friends. It doesn't really have any effect on the creatures, but it does tell you how they feel about you.

Lines 16000–17000 make up the SAY command, which give you an alternate way of casting spells, and little else. Some adventures use

—◉→

AMPERWARE ???

You're Kidding? Not Another Amper-Utility!

Yes folks, another amper-utility; but not *just* another amper-utility. Amperware is a very special extension to Applesoft* which can make your application program rival the best commercially available. You get all the power of assembly language without worrying about registers, flags, stacks and similar nonsense. Just add three lines to your HELLO program and Applesoft* is immediately extended to include many new commands like:

& INPUT - create powerful video forms
& FIND - fast string array searching
& SORT - rapid array sorts
& READ/WRITE - quick disk access in binary (up to 16 x faster!)
& PRINT WITH - complete formatted printing

Amperware was written with the novice programmer in mind. No PEEKs, No POKEs, No CALLs . . . anyone can use it. The assembly programmer will find it a good starting point which can be extended as needed. Whatever your skill level, we think you'll like Amperware. At this price you can't go wrong.

ONLY $49.95** (+ $1.50 shipping)

(Indiana Residents Add 5% Sales Tax)

Ask at your local dealer or order directly from:

Scientific Software Products, Inc.

3171 Donald Avenue
Indianapolis, In 46224
(317) 299-0467

*Applesoft is a registered trademark of Apple Computer Inc.
**Master Card and VISA accepted.

SSP inc.

Circle 13 on Reader Service Card

magic words, so this routine is modified to let you open doors by typing "SAY OPEN SESAME."

Lines 17000–18000 let you READY a weapon. You can carry a huge number of weapons, but you can only fight with one at a time. READYing a weapon is how you specify which one you're using.

Lines 18000–19000 are the "Save Game" routine, added to the program by John Nelson. The pointers to the start and end of variable and array space are saved into a binary file, and then the variable and array spaces are saved into two more binary files.

Lines 19000–20000 restore a saved game. By retrieving the variables and variable pointers, the game begins exactly where it left off.

Lines 50000–60000 are the error-handling routines, consisting of a short machine-language program that helps fix APPLESOFT's ON-ERR GOTO bugs, a routine to print out the error message (which is disabled by ONERR GOTO), and a jump back to 100. This doesn't always work, but it's helpful. If you are running an EAMON adventure that doesn't have the error-trapping routine, you can get back to where you left off by typing

POKE 51,0 : GOTO 100

The Base Program is shown in Listing 2. Table 1 gives an explanation of the variables in the program.

The basic instructions for playing EAMON are given in Listing 1, and tell a lot about the game, especially the combat details.

The Dungeon Designer Disk contains a number of programs that help you to create adventures. The most important of these, DUNGEON EDIT, is used to create the descriptions of rooms, monsters, and artifacts. There are several other useful programs, as well as the *Players' Manual* and the *Dungeon Designers' Manual* (which come on text files, with a program to print them out to the screen or a printer).

Getting EAMON. There are two good sources of EAMON adventures. One is good old *A.P.P.L.E.;* the other is the Apple Avocation Alliance, Inc.

A.P.P.L.E. has a number of adventures available on its "As Is Software" disks, which cost $4.00 each.

The Apple Avocation Alliance is in business to do essentially the same job as *A.P.P.L.E.'s* "As Is Software" — the distribution of public-domain software at low prices. Ron Maleika runs 3A almost single-handedly, and although he has complained about being swamped with orders recently, service has consistently been very fast — my orders have been filled from six days to two weeks from the time of mailing them in. *A.P.P.L.E.* averages about four weeks.

3A will ship disks on DOS 3.2 or 3.3, and Ron is actively acquiring all the EAMON adventures in the known universe. Not a bad deal. Table 2 gives a list of the EAMON adventures in 3A's October, 1982 catalog, plus two more that Ron told me about later.

If you are not an *A.P.P.L.E.* member, you may order direct from:

Apple Avocation Alliance, INC.
721 Pike Street
Cheyenne, Wyoming USA 82009

and enclose $2.00 for a catalog, and $3.00 for year's subscription. 3A has a huge number of public-domain programs which they sell for $1.00 plus the price of the disk (currently at $2.15). 3A has over thirty EAMON adventures, including the *EAMON Master Disk* (which you *must* have to run the adventures), the *Dungeon Designer Disk,* and two *EAMON Utility Disks.*

Conclusions.

A lot of the features in EAMON are inelegant, and some are downright crude. Still, EAMON makes a very good system for adventure games, especially combat-oriented adventures. Since it's in the public domain, anyone can use it and try to improve it — but in its present form it is eminently playable and very enjoyable.

The BASE PROGRAM shows you one way to write an adventure game, and makes a good starting point for those who want to write their own adventure systems. People tend to start out by just playing the adventures, but eventually they can't resist the temptation to write one. Before they know it, they're writing their own adventure system.

Whether you're one of the fanatics described above, or just like to play adventure games, EAMON is worth looking into. The price is right, the variety is stunning, and the quality is surprisingly high.

Happy adventuring!

The EAMON Adventures:

01	EAMON Master Disk and The Beginners' Cave
02	Minotaur's Lair
03	Caves of the Mind
04	Zyphur River Venture
05	Doom Castle
06	The Death Star
07	The Devil's Tomb
08	The Abductor's Quarters
09	Assault on the Clone Master
10	The Magic Kingdom
11	Molinar's Tomb
12	The Quest for Trezore
13	Treasure Island
14	Furioso
15	Heroes Castle
16	The Caves of Mondamen
17	Merlin's Castle
18	Hogarth Castle
19	The Death Trap
20	The Black Death
21	Marron Quest
22	The Senator's Chambers
23	The Temple of Ngurct
24	Black's Mountain
25	Nuclear Nightmare
26	Moleman Assault
27	Moleman Revenge
28	London Tower
29	The Lost Island of Apple
30	The Underground City
31	Gauntlet
32	House of Ill Repute
33	Nobbin's Hell Hole

EAMON Tournament Adventures:

60	Castle of Count Fuey
61	Search for the Key
62	The Rescue Mission

Utilities:

Dungeon Designers' Disk
EAMON Utilities I
EAMON Utilities II

Table I
Variables used in EAMON's
BASE DUNGEON PROGRAM
from A Manual for EAMON Adventure Designers
by Donald Brown

AC –	ARMOUR CLASS of player character (how many points of damage the character's armour will absorb.
AD%(*,*) –	ARTIFACT DATA The first subscript is the number of the artifact, and the key for the seconds is (1 = Value, 2 = Type, 3 = Weight, 4 = Room, 5 = Complexity, 6 = Type, 7 = Dice, 8 = Sides, 9 = Flag is artifact has been seen before)
AE –	ARMOUR EXPERTISE
AN$(*) –	NAMES OF ARTIFACTS
BANK –	GOLD PLAYER HAS IN BANK
C –	HOLDS NUMBER OF COMMAND GIVEN
C$(*) –	VERBS PROGRAM RESPONDS TO
CH –	PLAYER CHARISMA
CA$ –	LAST COMMAND GIVEN
DF –	DEFENDER
DIE –	LOGICAL FLAG, 1 = PLAYER HAS DIED
DK$ –	HOLDS CTRL-D FOR DISK COMMANDS
DR%(*) –	ROOM MOVED IN EACH DIRECTION
EA –	EFFECT OF ARMOUR ON ODDS-TO-HIT
FD%(*) –	FULL DAMAGE OF A SIDE IN COMBAT
FR –	FUMBLE ROLL or FRIEND RATING
GOLD –	GOLD PLAYER HAS ON PERSON
HIT –	LOGICAL FLAG IF HIT IN COMBAT
INC –	LOGICAL FLAG IF ABILITY INCREASED
LK –	LOGICAL FLAG IF "LOOKED" ALREADY
MD%(*,*) –	MONSTER DATA First subscript is monster number, second key is (1 = HD, 2 = AG, 3 = FRIEND, 4 = COUR., 5 = ROOM, 6 = WGHT, 7 = DEF ODDS, 8 = ARMOUR, 9 = WEAPON #, 10 = ODDS TO HIT, 11 = W DICE, 12 = W SIDES, 13 = HITS TAKEN, 14 = REACTION; 0 = NOT MET, 1 = HOSTILE, 2 = NEUTRAL, 3 = FRIENDLY)
MN$(*) –	NAME OF MONSTER
MR –	MONSTER MORALE
NA –	NUMBER OF ARTIFACTS
NBTL –	LOGICAL FLAG IF IN BATTLE
NC –	NUMBER OF COMMANDS
NM :	NUMBER OF MONSTERS
NW –	TOTAL COUNT OF WEAPONS IN GAME
NZ :	NUMBER ARTIFACTS BESIDES THE PLAYER'S WEAPONS
LOF –	NUMBER OF OFFENSIVE (ATTACKING) MONSTER
RAISE –	LOGICAL FLAG IF POWER RAISED SOMEONE FROM THE DEAD
REC –	PLAYER RECORD IN CHARACTERS FILE
RL –	RANDOM NUMBER 1-100
ROOM –	ROOM PLAYER IS IN
RR –	RANDOM NUMBER 1-100 FOR POWER SPELL
S$ –	SUBJECT OF COMMAND GIVEN
S2%(*) –	CURRENT SPELL ABILITY
SA%(*) –	TOTAL SPELL ABILITY
SEX$ –	HOLDS "M" OR "F" FOR PLAYER
SPD –	NUMBER OF TURNS SPEED SPELL HAS TO GO
SUC –	LOGICAL FLAG IF SPELL SUCCEEDED
TD%(*) –	DAMAGE TAKEN IN BATTLE FOR SIDE
TP –	TOTAL PRICE OF TREASURE
V$ –	VERB OF COMMAND
V%(*) –	FLAGS IF PLAYER HAS BEEN IN ROOM
WA%(*) –	PLAYER'S WEAPON ABILITY
WD%(*) –	FOR WEAPON, DICE OF DAMAGE
WN%(*) –	NAME OF PLAYER'S WEAPON
WO%(*) –	WEAPON COMPLEXITY
WP%(*) –	WEAPON POINTER (USED AT THE END OF GAME)
WS%(*) –	SIDES/DIE OF DAMAGE FOR WEAPON
WT –	WEIGHT PLAYER CARRYING
WT%(*) –	WEAPON TYPE
WZ –	NUMBER OF WEAPONS PLAYER BROUGHT

Listing 1

You read the instructions and they say —

Information About THE WORLD OF EAMON

You will have to buy a weapon. Your chance to hit with it will be the weapon complexity, plus your ability in that class, plus twice your agility.

The five classes of weapons (and your current abilities with each) are —

Club/Mace	20%
Spear	10%
Axe	5%
Sword	0%
Bow	-10%

Every time you score a hit in battle, your ability in the weapon class may go up by 2%, if a random number from 1-100 is less than your chance to miss!

There are four armour types, and you may also carry a shield if you do not use a two-handed weapon. These protections will absorb hits placed upon you (almost always!) but they lower your chance to hit. The protections are —

—◉→

Armour	Hits Protect	Odds Adjust
None	0	- 0%
Leather	1	-10%
Chain	2	-20%
Plate	5	-60%
Shield	1	- 5%

You will develop an armour expertise, which will go up when you hit a blow wearing armour and your expertise is less than the armour you are wearing. No matter how high your armour expertise is, however, the net effect of armour will never increase your chance to hit.

You can carry weights up to ten times your hardiness, or 150 gronds. (A measure of weight, one grond = 10 DOS.)

Additionally, your hardiness tells how many points of damage you can survive. Therefore, you can be hit with 15 1-point blows before you die.

However, you will not be told how many blows you have taken. You will be merely told things such as —

"Wow! That one hurt!"
or "You don't feel very well."

Your charisma (20) affects how citizens of EAMON react to you. You affect a monster's friendliness rating by your charisma less ten, difference times two (20%).

You start off with 200 gold pieces, which you will want to spend on supplies for your first adventure. You will get a lower price for items if your charisma is high.

After you begin to accumulate wealth, you may want to put some of your money into the bank, where it cannot be stolen. However, it is a good idea to carry some gold with you for use in bargaining and ransom situations.

You may also hire a wizard to teach you some magic spells. There are four spells you may learn.

Blast	—Hurt your enemies from a distance.
Heal	—Remove damage from your body.
Speed	—Double your dexterity for a time.
Power	—Does something weird. The exact effect is unpredictable.

Other types of magic may work in various adventures and items may have special properties. However, these will not work in other adventures than where they were found. Thus, it is best (and you have no choice but to) sell all items found in adventures, except for weapons and armour.

The man behind the desk takes back the instructions and says, "It is now time for you to start your life." He makes an odd sign with his hand and says, "Live long and prosper."

You now wander into the main hall.

```
]LIST

5  REM

        EAMON ADVENTURE
       BASE PROGRAM  2.1

        BY DONALD BROWN

     WITH MODIFICATIONS  BY
JOHN NELSON & ROBERT PLAMONDON

CALL -A.P.P.L.E. : MARCH 1983

10  ONERR  GOTO 50
15  REM
*** IF 'GAME.PTRS' EXISTS
     , A GAME HAS BEEN SAVED.
16  REM *** IF IT DOESN'T EXIST,
     THE ERROR WILL CAUSE A JUMP
     TO 50.
20 DK$ =  CHR$ (4): PRINT DK$"VER
    IFY GAME.PTRS": GOTO 19000
50  POKE 216,0: GOTO 1000: REM **
    * SET UP THE GAME.
100  REM
**** MAIN LOOP. GIVE RO
    OM DESC, THEN GET COMMAND.

105  SPEED= 255
110  PRINT
120  IF SPD THEN SPD = SPD - 1: IF
     NOT SPD THEN MD%(0,2) = MD%
    (0,2) / 2:MD%(0,10) = MD%(0,
    10) - 2 * MD%(0,2): IF  RND
    (1) > .8 THEN  PRINT "YOUR S
    PEED SPELL HAS JUST EXPIRED!
    ": PRINT
128  REM
*** PRINT ROOM DESCRIPTI
    ON. IF ROOM HASN'T BEEN SEEN
     BEFORE.
129  REM  *** GIVE A DESCRIPTION.
     OTHERWISE JUST PRINT THE RO
    OM NAME.
130  IF V%(ROOM) THEN  PRINT "YOU
     ARE": PRINT " ";RN$(ROOM): GOTO
    150
140  PRINT DK$;"READ EAMON.DESC.R
    ";ROOM: INPUT A$: PRINT DK$:
     PRINT A$: PRINT :V%(ROOM) =
    1
149  REM
*** PRINT OUT NAMES OR
    DESCRIPTIONS OF MONSTERS IN
    ROOM.
150  FOR M = 1 TO NM: IF MD%(M,5)
     <  > ROOM THEN  NEXT : GOTO
    170
155  IF MD%(M,15) THEN  PRINT "
     >> ";MN$(M);" IS HERE.": NEXT
    : GOTO 170
160  PRINT DK$;"READ EAMON.DESC.R
    ";M + 300: INPUT A$: PRINT D
    K$:MD%(M,15) = 1: IF A$ <  >
    "" THEN  PRINT A$: PRINT : NEXT

169  REM
*** PRINT OUT NAMES OR
    DESCRIPTIONS OF ARTIFACTS IN
     ROOM.
170  FOR A = 1 TO NZ: IF AD%(A,4)
     <  > ROOM THEN  NEXT : GOTO
    190
```

```
175  IF AD%(A,9) THEN  PRINT "
     -- YOU SEE ";AN$(A): NEXT : GOTO
     190
180  PRINT DK$;"READ EAMON.DESC,R
     ";A + 100: INPUT A$: PRINT D
     K$: PRINT A$:AD%(A,9) = 1: PRINT
     : NEXT
189  REM
*** PRINT OUT NAMES OF
     PLAYER CHARACTER'S DROPPED W
     EAPONS.
190  FOR A = A TO NA: IF AD%(A,4)
      = ROOM THEN  PRINT "   -> Y
     OUR ";AN$(A);" IS HERE."
200  NEXT A:V%(ROOM) = 1
205  REM
***** GET COMMAND *****


210  INVERSE : PRINT "YOUR COMMAN
     D?";: NORMAL : INPUT " ";A$
218  REM
*** PARSE INPUT:  GET R
     ID OF LEADING SPACES.
220  IF  LEFT$ (A$,1) = " " THEN
     A$ =  MID$ (A$,2): GOTO 220
229  REM
*** IF INPUT IS A NULL
     STRING, DO PREVIOUS COMMAND
     AGAIN.
230  IF A$ = "" THEN A$ = CZ$: VTAB
      PEEK (37): HTAB 17: PRINT A
     $: GOTO 275
239  REM
*** FIND END OF FIRST W
     ORD (THE VERB) BY LOOKING FO
     R A SPACE
240 CZ$ = A$: FOR A = 2 TO  LEN (
     A$): IF  MID$ (A$,A,1) (  )
     " " THEN  NEXT
249  REM
*** FIRST WORD IS VERB
     (V$), SECOND WORD IS SUBJECT
      (S$).
250 V$ =  LEFT$ (A$,A - 1):S$ =  MID$
    .(A$,A + 1)
255 LV =  LEN (V$)
260  IF  LEFT$ (S$,1) = " " THEN
     S$ =  MID$ (S$,2)  GOTO 260
269  REM
**** SEARCH THROUGH LIS
     T OF VERBS FOR MATCH WITH CO
     MMAND ***

270  FOR C = 1 TO 6: IF  LEFT$ (C
     $(C),LV) = V$ THEN 290
272  NEXT :F = 0: FOR A = 7 TO NC
     : IF C$(A) = V$ THEN C = A: GOTO
     290
273  IF  LEFT$ (C$(A),LV) = V$ THEN
     F = F + 1:C = A: IF F ) 1 THEN
     275
274  NEXT : IF F THEN 290
275  PRINT : PRINT "HUH? I ONLY U
     NDERSTAND THESE COMMANDS--
        ";: FOR C = 1 TO NC: PRINT
     C$(C); SPC( 20 -  LEN (C$(C)
     ));: NEXT : PRINT : PRINT : GOTO
     210
280  IF C$(C) = S$ THEN S$ = V$:V
     $ = C$(C)
285  SPEED= 200
289  REM
***** JUMP TO ROUTINE T
     HAT HANDLES COMMAND *****

290  ON C GOTO 3000,3000,3000,300
     0,3000,3000,4000,4000,5000,6
     000,6000,7000,8000,9000,1000
     0,11000,12000,13000,14000,15
     000,15000,16000,17000,18000
300  REM
***** COMMANDS RETURN T
     O HERE:  COMBAT LOOP *****

309  REM
*** IF NOT IN BATTLE, G
     OTO 500
310  IF  NOT NBTL THEN 500
319  REM
*** DO COMBAT FOR EACH
     MONSTER IN ROOM.
320  FOR M = 1 TO NM: IF MD%(M,5)
      (  ) ROOM THEN 490
329  REM
*** MORALE CHECK.  SEE
     IF MONSTER TURNS TAIL AND RU
     NS.
330 M2 = MD%(M,14) - (MD%(M,14) =
     2):MR = 100 * FD%(M2) / TD%(
     M2) +  INT (41 *  RND (1) -
     20) - 20 * (MD%(M,9) =  - 1)

340  IF MD%(M,4) ( MR THEN  INVERSE
     : PRINT MN$(M);" FLEES OUT A
     N EXIT.": NORMAL : GOSUB 850
     0:MD%(M,5) = R2:M2 = MD%(M,1
     4):TD%(M2) = TD%(M2) - MD%(M
     ,1):FD%(MD) = FD%(M2) - MD%(
     M,13): GOTO 490
344  REM
*** BRANCH ACCORDING TO
     MONSTER REACTION: 1=HOSTILE,
      2=NEUTRAL, 3=FRIENDLY ***
345  ON MD%(M,14) GOTO 360,490,39
     0
359  REM
*** MONSTER IS HOSTILE.
       HAVE HIM ATTACK SOMEONE ON
      OUR SIDE ***

360 OF = M: IF TD%(3) = MD%(0,1) THEN
     DF = 0: GOSUB 7500: GOTO 490

370  FOR M2 = 1 TO NM: IF MD%(M2,
     5) = ROOM AND MD%(M2,14) = 3
      AND  RND (1) ( .25 THEN DF =
     M2: GOSUB 7500:M2 = 200: NEXT
     M2: GOTO 490
380  NEXT M2:DF = 0: GOSUB 7500: GOTO
     490
389  REM
*** MONSTER IS FRIENDLY
     .  HAVE HIM ATTACK AN ENEMY
     ***

390 OF = M
400  FOR M2 = 1 TO NM: IF MD%(M2,
     5) = ROOM AND MD%(M2,14) = 1
      THEN DF = M2: GOSUB 7500:M2
      = 200: NEXT M2: GOTO 490
410  NEXT M2
489  REM
*** IF ANY FOES ARE LEF
     T ALIVE, WE'RE STILL IN BATT
     LE.
490 NBTL = (FD%(1) ( TD%(1)): IF
     NBTL THEN  NEXT M
500  REM
***** LOOP TO LINE 100
     FOR NEXT COMMAND *****

900  GOTO 100
```

EAMON ADVENTURE

```
1000  REM
****** INITIALIZATION
      ROUTINE ******

1005  REM
*** GET NUMBER OF ROOM
      S, ARTIFACTS, EFFECTS, AND M
      ONSTERS.
1010  PRINT DK$;"OPEN EAMON.DESC"
      : PRINT DK$;"READ EAMON.DESC
      ": INPUT NR,NZ,NE,NM: PRINT
      DK$;"CLOSE":NA = NZ
1019  REM
*** READ IN ARTIFACT D
      ATA.
1020  PRINT DK$;"OPEN EAMON.ARTIF
      ACTS.L128": DIM AN$(NA + 4),
      AD%(NA + 4,9): FOR A = 1 TO
      NA: PRINT DK$;"READ EAMON.AR
      TIFACTS,R";A: INPUT AN$(A): FOR
      A2 = 1 TO 4: INPUT AD%(A,A2)
      : NEXT A2: IF AD%(A,2) ) 1 THEN
       FOR A2 = 5 TO 8: INPUT AD%(
      A,A2): NEXT A2
1029  REM
*** READ IN MONSTER DA
      TA.
1030  NEXT A: PRINT DK$;"OPEN EAM
      ON.MONSTERS.L128": DIM MN$(N
      M),MD%(NM,15): FOR A = 1 TO
      NM: PRINT DK$;"READ EAMON.MO
      NSTERS,R";A: INPUT MN$(A): FOR
      A2 = 1 TO 12: INPUT MD%(A,A2
      ): NEXT A2,A: PRINT DK$;"CLO
      SE"
1039  REM
*** READ IN CHARACTER
      DATA
1040  PRINT DK$;"OPEN FRESH MEAT"
      : PRINT DK$;"READ FRESH MEAT
      ": INPUT REC: INPUT MN$(0),M
      D%(0,1),MD%(0,2),CH: FOR A =
      1 TO 4: INPUT SA%(A): NEXT :
       FOR A = 1 TO 5: INPUT WA%(A
      ): NEXT : INPUT AE,SEX$,GOLD
      ,BANK,AC
1049  REM
*** READ IN DATA FOR C
      HARACTER'S WEAPONS.
1050  NW = 15: DIM WN$(NW),WT%(NW)
      ,WO%(NW),WD%(NW),WP%(NW): FOR
      A = 1 TO 4: INPUT WN$(A),WT%
      (A),WO%(A),WD%(A),WS%(A): NEXT
      : PRINT DK$;"CLOSE"
1060  PRINT DK$;"OPEN EAMON.DESC,
      L256": PRINT DK$;"OPEN EAMON
      .ROOM NAMES.L64": PRINT DK$;
      "OPEN EAMON.ROOMS,L64"
1069  REM
*** PLAY WITH WEAPON N
      AMES TO ASSURE MONSTER WEAPO
      NS DON'T HAVE THE SAME NAMES
       AS CHARACTER WEAPONS.
1070  FOR W2 = 1 TO 4: IF WN$(W2)
      (  ) "NONE" THEN NA = NA +
      1:AN$(NA) = WN$(W2):AD%(NA,2
      ) = 2:AD%(NA,3) = 2:AD%(NA,4
      ) =  - 1:AD%(NA,5) = WO%(W2)
      :AD%(NA,6) = WT%(W2):AD%(NA,
      7) = WD%(W2):AD%(NA,8) = WS%
      (W2): NEXT :W2 = 4
1080  WZ = W2:WT = W * WZ
1090  FOR W2 = 1 TO WZ: FOR A = 1
      TO NA - WZ: IF AN$(A) = WN$
      (W2) THEN AN$(A) = AN$(A) +
      "*": FOR A = 0 TO 0: NEXT : GOTO
      1090
1100  NEXT A,W2:EA = 0:MD%(0,8) =
      0:A2 =  INT (AC / 2): IF A2 *
      2 (  ) AC THEN MD%(0,8) = 1:
      EA =  - 5
1110  IF A2 THEN MD%(0,8) = MD%(0
      ,8) + A2:EA = EA - A2 * 10: IF
      A2 = 3 THEN MD%(0,8) = MD%(0
      ,8) + 2:EA = EA - 30
1120  IF NA ) NZ THEN MD%(0,10) =
      (EA + AE) * ( - EA ) AE) + W
      O%(1) + WA%(WT%(1)) + 2 * MD
      %(0,2):MD%(0,7) = 0:MD%(0,9)
       = NZ + 1:MD%(0,11) = WD%(1)
      :MD%(0,12) = WS%(1):MD%(0,14
      ) = 3
1129  REM
*** READ COMMAND WORDS
       INTO AN ARRAY
1130  READ NC: DIM C$(NC): FOR C =
      1 TO NC: READ C$(C): NEXT
1140  FOR S = 1 TO 4:S2%(S) = SA%
      (S): NEXT : DIM V%(96):ROOM =
      1
1199  REM
*** READ IN ROOM NAMES
       AND DATA ON WHICH ROOM CONN
      ECTS WHERE.
1200  DIM ED%(NR,6)
1210  FOR X = 1 TO NR: PRINT DK$;
      "READ EAMON.ROOMS,R";X: FOR
      Y = 1 TO 6: INPUT ED%(X,Y): NEXT
      Y,X
1230  DIM RN$(NR)
1250  FOR X = 1 TO NR: PRINT DK$;
      "READ EAMON.ROOM NAMES,R";X:
       INPUT RN$(X): NEXT
1299  REM
*** RANDOM NUMBER FUNC
      TION TO RETURN INTEGER FROM
      1 TO 100.
1300  DEF  FN R(X) =  INT ( RND (
      1) * 100 + 1)
1349  REM
*** ERROR-HANDLING ROU
      TINE IS AT 59000
1350  ONERR  GOTO 59000
1899  REM
*** GOTO 100 TO START
      GAME.

1900  GOTO 100
1901  REM
*** DATA ***
1910  DATA     24
1920  DATA     NORTH,SOUTH,EAST,WE
      ST,UP,DOWN,GET,TAKE,DROP,LOO
      K,EXAMINE,ATTACK,FLEE,GIVE,I
      NVENTORY,BLAST,HEAL,POWER,SP
      EED,SMILE,WAVE,SAY,READY,SAV
      E
2000  REM
**** END GAME ROUTINE
      *****

2005  SPEED= 255
2010  PRINT : PRINT "(HIT ANY KEY
      TO CONTINUE)   ";: POKE  -
      16368,0: GET A$: PRINT
2020  PRINT DK$"CLOSE": IF SPD THEN
      MD%(0,2) = MD%(0,2) / 2
```

```
2029  REM
*** IF CHARACTER IS DE
    AD, SKIP THIS STUFF. ***
2030  IF DIE THEN 2500
2040  FOR W = 1 TO 4:WN$(W) = "NO
    NE": NEXT :W2 = 1: FOR A = 1
     TO NA: IF AD%(A,2) ) 1 AND
    AD%(A,4) =  - 1 THEN WN$(W2)
     = AN$(A):WP%(W2) = A:W2 = W
    2 + 1:AD%(A,4) = 0
2049  REM
*** CHECK FOR EXCESS W
    EAPONS.  MAKE HIM GET RID OF
     THEM. ***
2050  NEXT A: IF W2 ( 6 THEN 2300

2060  HOME : VTAB 5: PRINT "AS YO
    U START TO ENTER THE MAIN HA
    LL.": PRINT "LORD WILLIAM MI
    SSLEFIRE APPEARS AND": PRINT
    "TELLS YOU, 'YOU HAVE TOO MA
    NY WEAPONS": PRINT "TO KEEP
    THEM ALL--4 IS THE LEGAL LIM
    IT.": PRINT : PRINT "YOUR WE
    APONS ARE--"
2070  FOR W = 1 TO W2 - 1: PRINT
     SPC( 4);W;"--";WN$(W): NEXT
    : PRINT : PRINT "ENTER THE N
    UMBER OF THE WEAPON TO SELL
        (1-";W2 - 1;") "
2080  VTAB  PEEK (37): HTAB 23: INPUT
    A$:W5 =  VAL (A$): IF W5 ( 1
     OR W5 ) W2 - 1 OR W5 (  )  INT
    (W5) THEN 2080
2090 AD%(WP%(W5),4) =  - 1:WN$(W5
    ) = WN$(W2 - 1):WP%(W5) = WP
    %(W2 - 1):W2 = W2 - 1: IF W2
     ) 5 THEN 2060
2300  FOR W = 1 TO 4: IF WP%(W) THEN
    WT%(W) = AD%(WP%(W),6):WO%(W
    ) = AD%(WP%(W),5):WD%(W) = A
    D%(WP%(W),7):WS%(W) = AD%(WP
    %(W),8)
2309  REM
***** SELL TREASURE **
    ***

2310  NEXT W: HOME : VTAB 5: PRINT
    "AS YOU DELIVER YOUR TREASUR
    ES TO SAM": PRINT "SLICKER,
    THE LOCAL BUYER FOR SUCH": PRINT
    "THINGS, HE EXAMINES YOUR GO
    ODS AND": PRINT "PAYS YOU ";

2320 TP = 0: FOR A = 1 TO NA: IF
    AD%(A,4) (  )  - 1 THEN 2350

2330  IF AD%(A,2) (  ) 0 THEN AD%
    (A,1) = AD%(A,1) * CH / 10
2340 TP = TP + AD%(A,1)
2350  NEXT A: PRINT TP;" GOLD PIE
    CES.": PRINT
2360 GOLD = GOLD + TP: IF GOLD (
    0 THEN GOLD = 0
2370  PRINT : PRINT "(HIT ANY KEY
    TO CONTINUE)  ";: GET A$: PRINT

2499  REM
***** GO HOME *****

2500  ONERR  GOTO 2500: REM *** R
    ESTART HERE ON ERROR
2510  HOME : VTAB 5: PRINT "(INSE
    RT EAMON MASTER DISKETTE, TH
    EN": PRINT "   HIT THE 'C' K
    EY)  ";: POKE  - 16368,0
2520  GET A$: IF A$ (  ) "C" THEN
    2520
2530  PRINT A$

2540  IF DIE THEN  PRINT DK$;"OPE
    N THE ADVENTURER,S6,D1": PRINT
    DK$;"DELETE THE ADVENTURER":
     GOTO 2900
2549  REM
*** WRITE CURRENT CHAR
    ACTER STATISTICS TO MAIN CHA
    RACTER FILE ***
2550  PRINT DK$;"OPEN CHARACTERS,
    L150,S6,D1": PRINT DK$;"WRIT
    E CHARACTERS,R";REC: PRINT M
    N$(0): PRINT MD%(0,1): PRINT
    MD%(0,2): PRINT CH: FOR A =
    1 TO 4: PRINT SA%(A): NEXT
2560  FOR A = 1 TO 5: PRINT WA%(A
    ): NEXT : PRINT AE: PRINT SE
    X$: PRINT GOLD: PRINT BANK: PRINT
    AC: FOR A = 1 TO 4: PRINT WN
    $(A): PRINT WT%(A): PRINT WO
    %(A): PRINT WD%(A): PRINT WS
    %(A): NEXT
2569  REM
*** WRITE POINTER TO C
    HARACTER FILE ***
2570  PRINT DK$;"OPEN THE ADVENTU
    RER": PRINT DK$;"WRITE THE A
    DVENTURER": PRINT MN$(0): PRINT
    REC: PRINT DK$;"CLOSE"
2899  REM
***** RUN MAIN HALL PR
    OGRAM ON MASTER DISK *****

2900  PRINT DK$;"RUN MAIN HALL,S6
    ,D1"
3000  REM
***** MOVE COMMAND ***
    **

3010  IF NBTL AND S$ (  ) "FLEE" THEN
     PRINT : PRINT "YOU CAN'T DO
     THAT WITH UNFRIENDLIES
       ABOUT!": PRINT : GOTO 100

3019  REM
*** FIND DIRECTION PLA
    YER WANTS TO MOVE
3020  FOR D = 1 TO 6: IF  LEFT$ (
    V$,1) (  )  MID$ ("NSEWUD",D
    ,1) THEN  NEXT
3030 R2 = ED%(ROOM,D)
3039  REM
*** IF R2 IS ZERO, YOU
     CAN'T GO THAT WAY. IF R2 IS
     NEGATIVE, MOVING THAT DIREC
    TION IS A 'SPECIAL MOVE.'
3040  IF R2 ) 0 THEN 3500
3050  REM
*** SPECIAL MOVES.
3059  REM
*** GOING TO ROOM -99
    TAKES YOU HOME.
3060  IF R2 =  - 99 THEN  PRINT :
     PRINT "YOU SUCCESSFULLY RID
    E OFF INTO THE": PRINT "  SU
    NSET.": GOTO 2000
3070  IF  NOT R2 THEN 3490
3490  PRINT : PRINT "YOU CAN'T GO
     THAT WAY!":V%(ROOM) = 0: GOTO
    100
3499  REM
*** SUCCESSFUL MOVE. R
    2 IS NEW ROOM, R3 IS OLD ROO
    M.
3500 R3 = ROOM:ROOM = R2: GOSUB 3
    600
3590  GOTO 100
```

```
3600  REM
*** CHECK MONSTER REAC
    TION: FRIENDLY, NEUTRAL, OR
    HOSTILE.

3610 TD%(1) = 0:TD%(3) = MD%(0,1)
    :FD%(1) = 0:FD%(3) = MD%(0,1
    3)
3619  REM
*** DID MONSTER FOLLOW
     CHARACTER INTO ROOM? ***
3620  FOR M = 1 TO NM: IF MD%(M,5
    ) = R3 THEN  IF MD%(M,14) =
    3 OR (MD%(M,14) = 1 AND 200 *
     RND (1) < MD%(M,4)) THEN MD
    %(M,5) = ROOM: GOTO 3670
3630  IF MD%(M,5) <  > ROOM THEN
    3900
3639  REM
*** IF MONSTER HAS BEE
    N MET BEFORE, SKIP NEXT SECT
    ION. ***
3640  IF MD%(M,14) THEN 3670
3649  REM
*** FIND REACTION OF M
    ONSTER ***
3650 FR = MD%(M,3): IF FR AND FR <
     > 100 THEN FR = FR +  INT (
    (CH - 10) / 2)
3660 MD%(M,14) = 1: IF FR > (100 *
     RND (1)) THEN MD%(M,14) = 2
    : IF FR > (100 *  RND (1)) THEN
    MD%(M,14) = 3
3670 A = MD%(M,14):FD%(A) = FD%(A
    ) + MD%(M,13):TD%(A) = TD%(A
    ) + MD%(M,1)
3895  REM
*** LOOP UNTIL WE RUN
    OUT OF MONSTERS ***
3900  NEXT M:NBTL = (TD%(1) > 0):
     RETURN
4000  REM
***** GET COMMAND ****
    *

4010  GOSUB 4900
4020  IF S$ = "ALL" THEN 4160
4100  REM
*** IS ITEM HERE? ***
4120  FOR A = 1 TO NA: IF (AN$(A)
     <  > S$ AND  LEFT$ (AN$(A),
     LEN (S$)) <  > S$ AND  RIGHT$
    (AN$(A), LEN (S$)) <  > S$) OR
    AD%(A,4) <  > ROOM THEN  NEXT
    : PRINT : PRINT "I SEE NO ";
    S$;" HERE!": PRINT : GOTO 10
    0
4129  REM
*** IS IT TOO HEAVY? *
    **
4130  GOSUB 4200: IF WT + AD%(A,3
    ) > 10 * MD%(0,1) THEN  PRINT
    : PRINT "IT IS TOO HEAVY FOR
     YOU.": PRINT : GOTO 100
4139  REM
*** GOT IT. ***
4140  PRINT : PRINT "GOT IT.":AD%
    (A,4) =  - 1:WT = WT + AD%(A
    ,3): PRINT : IF AD%(A,2) < 2
     OR MD%(0,9) <  > - 1 THEN
    300
4149  REM
*** IF ITEM IS A WEAPO
    N, AND CHARACTER IS UNARMED,
     READY THE WEAPON. ***
4150  GOTO 17000
4159  REM
*** GET ALL COMMAND **

4160  FOR A = 1 TO NA: IF AD%(A,4
    ) <  > ROOM THEN 4190
4170  GOSUB 4200: IF WT + AD%(A,3
    ) > 10 * MD%(0,1) THEN  PRINT
    AN$(A);" IS TOO HEAVY": GOTO
    4190
4180  PRINT AN$(A) + " TAKEN.":AD
    %(A,4) =  - 1:WT = WT + AD%(
    A,3)
4190  NEXT : PRINT : GOTO 300
4200  REM
*** INSERT SPECIAL EFF
    ECTS OF PICKING SOMETHING UP
     HERE ***
4210  RETURN
4900  REM
*** ROUTINE TO GET S$
    IF NOT PREVIOUSLY SPECIFIED.
     ***
4905  IF S$ = "" THEN  PRINT : PRINT
    C$(C);" WHAT";: INPUT S$: GOTO
    4900
4910  RETURN
5000  REM
***** DROP COMMAND ***
    **

5010  GOSUB 4900
5020  IF S$ = "ALL" THEN 5100
5029  REM
*** DOES HE HAVE IT? *
    **
5030  FOR A = 1 TO NA: IF (AN$(A)
     <  > S$ AND  LEFT$ (AN$(A),
     LEN (S$)) <  > S$ AND  RIGHT$
    (AN$(A), LEN (S$)) <  > S$) OR
    AD%(A,4) <  > - 1 THEN  NEXT
    : PRINT : PRINT "YOU AREN'T
    CARRYING A ";S$: PRINT : GOTO
    100
5039  REM
*** DROP IT. ***
5040 WT = WT - AD%(A,3):AD%(A,4) =
    ROOM: PRINT AN$(A);" DROPPED
    .": PRINT
5049  REM
*** TOO MUCH WEIGHT RO
    UTINE ***
5050  IF WT > 10 * MD%(0,1) THEN
     PRINT "YOU SUDDENLY FIND YO
    U CANNOT CARRY ALL  OF THE I
    TEMS YOU ARE CARRYING. AND T
    HEY ALL FALL TO THE GROUND."
    : PRINT : GOTO 5100
5060  IF MD%(0,9) > 0 THEN  IF AD
    %(MD%(0,9),4) <  > - 1 THEN
    MD%(0,9) =  - 1
5070  GOTO 300
5099  REM
*** DROP ALL ROUTINE *
    **
5100  FOR A = 1 TO NA: IF AD%(A,4
    ) =  - 1 THEN  PRINT AN$(A);
    " DROPPED.":AD%(A,4) = ROOM
5110  NEXT A:WT = 0: GOTO 5060
6000  REM
***** LOOK COMMAND ***
    **
```

```
6019  REM
*** GIVE ARTIFACT DESC
    RIPTION ***
6020 LK = 0: FOR A = 1 TO NZ: IF
    AN$(A) = S$ AND (AD%(A,4) =
    ROOM OR AD%(A,4) =  - 1) THEN
     PRINT DK$;"READ EAMON.DESC.
    R";A + 100: INPUT A$: PRINT
    DK$: PRINT : PRINT A$: PRINT
    :LK = 1
6030  NEXT A: IF LK THEN 300
6039  REM
*** GIVE MONSTER DESCR
    IPTION ***
6040  FOR A = 1 TO NM: IF MN$(A) =
    S$ AND MD%(A,5) = ROOM THEN
     PRINT DK$;"READ EAMON.DESC,
    R";A + 300: INPUT A$: PRINT
    DK$: PRINT : PRINT A$: PRINT
    :LK = 1
6050  NEXT A: IF LK THEN 300
6059  REM
*** SET FLAG SO ROOM D
    ESCRIPTION WILL BE GIVEN IN
    MAIN LOOP ***
6060 V%(ROOM) = 0: GOTO 300
7000  REM
********** ATTACK COMM
    AND **********

7010  GOSUB 4900
7279  REM
*** MAKE SURE VICTIM I
    S HERE. ***
7280  FOR M = 1 TO NM: IF (S$ (  )
    MN$(M) AND  LEFT$ (MN$(M), LEN
    (S$)) (  ) S$ AND  RIGHT$ (M
    N$(M), LEN (S$)) (  ) S$) OR
    MD%(M,5) (  ) ROOM THEN  NEXT
    : PRINT : PRINT "ATTACK WHO?
    ": PRINT : GOTO 100
7289  REM
*** IS ATTACKER ARMED?
     ***
7290  IF MD%(0,9) =  - 1 THEN  PRINT
    : PRINT "YOU HAVE NO WEAPON
    READY!": PRINT : GOTO 100
7299  REM
**** DO THE ATTACK. IF
     IT DOESN'T WORK, SKIP.****
7300 OF = 0:DF = M: GOSUB 7500: IF
     NOT HIT THEN 7360
7309  REM
*** CHECK FOR SKILL IN
    CREASE. ***
7310  IF INC THEN W2 = AD%(MD%(0,
    9),6):WA%(W2) = WA%(W2) + 2
7319  REM
*** CHECK FOR INCREASE
     IN ARMOR EXPERTISE. ***
7320  IF  - EA (  = AE THEN 300
7330  IF  INT (100 *  RND (1) + 1
    ) (  = MD%(0,10) THEN 7360
7340 A = 2: IF EA + AE =  - 1 THEN
    A = 1
7350 AE = AE + A:MD%(0,10) = MD%(
    0,10) + A
7360  IF MD%(DF,14) = 1 OR DF = 0
     THEN 300
7369  REM
*** ATTACKING A MONSTE
    R DECREASES ITS FRIENDLINESS
     ***
7370 MD%(DF,3) = MD%(DF,3) / 2: FOR
    M = 1 TO NM: IF MD%(M,5) = R
    OOM AND MD%(M,14) ) 1 THEN M
    D%(M,14) = 0
7380  NEXT M:R3 = ROOM: GOSUB 360
    0: GOTO 300

7400  REM
**** MONSTER TRIES TO
     PICK UP A WEAPON ****
7410  IF MD%(OF,10) = 0 THEN  RETURN

7419  REM
*** SEE IF THERE'S A W
    EAPON ON THE FLOOR ***
7420  FOR A = 1 TO NA: IF AD%(A,4
    ) (  ) ROOM OR AD%(A,2) ( 2 THEN
     NEXT : RETURN
7430  PRINT MN$(OF);" PICKS UP ";
    AN$(A): PRINT :AD%(A,4) = 0:
    MD%(OF,9) = A:MD%(OF,10) = M
    D%(OF,10) + AD%(A,5):MD%(OF,
    4) = MD%(OF,4) * 2: RETURN
7499  REM
***** ATTACK ROUTINE *
     ****

7500 HIT = 0: IF MD%(OF,9) =  - 1
     THEN 7400
7504  REM
*** PRINT ATTACK MESSA
    GE AND ROLL THE 'DICE'. ***
7505  PRINT MN$(OF);" ATTACKS ";M
    N$(DF): PRINT " --";:RL =  INT
    (100 *  RND (1) + 1):HIT = 0
    : IF (RL ( 5 OR RL ( MD%(OF,
    10) - MD%(DF,7)) AND RL ( 96
     THEN HIT = 1
7510  IF HIT THEN 7600
7519  REM
*** A MISS.  IS IT A F
    UMBLE? ***
7520  IF RL ( 97 THEN  PRINT "A M
    ISS.": RETURN
7525  INVERSE : PRINT "A FUMBLE!"
    : NORMAL :FR =  FN R(X)
7527  IF FR ( 35 + 40 * ( NOT MD%
    (OF,9)) THEN  PRINT "  FUMB
    LE RECOVERED.": PRINT : RETURN

7530  IF FR ( 76 THEN  PRINT "  W
    EAPON DROPPED!":AD%(MD%(OF,9
    ),4) = ROOM:MD%(OF,10) = MD%
    (OF,10) - AD%(MD%(OF,9),5):M
    D%(OF,9) =  - 1:MD%(OF,4) =
    MD%(OF,4) / 2: PRINT : RETURN

7532  IF MD%(OF,9) = 0 GOTO 7550
7535  IF FR ) 95 THEN 7550
7536  REM
*** HIGH-QUALITY WEAPO
    NS BREAK LESS OFTEN. ***
7537  IF AD%(MD%(OF,9),5) * 3 (  FN
    R(X) THEN  PRINT "  FUMBLE
    RECOVERED.": PRINT : RETURN

7540  PRINT "  WEAPON BROKEN!":AD
    %(MD%(OF,9),4) = 0:MD%(OF,10
    ) = MD%(OF,10) - AD%(MD%(OF,
    9),5):MD%(OF,9) =  - 1:MD%(O
    F,4) = MD%(OF,4) / 2: IF  RND
    (1) ) .5 THEN  PRINT : RETURN

7545  PRINT "  BROKEN WEAPON HURT
    S USER!"
7550  IF FR = 100 THEN 7560
7555 D = MD%(OF,11):S = MD%(OF,12
    ):DF = OF:A = 1: GOTO 7635
7560 D = MD%(OF,11) * 2:S = MD%(O
    F,12):DF = OF:A = 0: GOTO 76
    35
7599  REM
*** A HIT!  CHECK FOR
    SKILL INCREASE ***
```

```
7600 INC = ( INT (100 *  RND (1) +
     1) ) MD%(OF,10)):MD%(OF,10) =
     MD%(OF,10) + 2 * INC
7605 D = MD%(OF,11):S = MD%(OF,12
     ):A = 1: IF RL ) 5 THEN  PRINT
     "A HIT!": GOTO 7635
7609  REM
*** CRITICAL HIT ROUTI
     NE.  CALCULATE EXTRA DAMAGE
     ***
7610  INVERSE : PRINT "A CRITICAL
     HIT!": NORMAL :R2 =  FN R(X
     ): IF R2 ( 51 THEN A = 0: GOTO
     7635
7615  IF R2 ( 86 THEN S = 1.5 * S
     : GOTO 7635
7620  IF R2 ( 96 THEN D = 2 * D: GOTO
     7635
7625  IF R2 ( 100 THEN D = 3 * D:
     GOTO 7635
7630  GOTO 7700
7634  REM
*** CALCULATE DAMAGE,
     SUBTRACT ARMOR VALUE FROM DA
     MAGE  ***
7635 D2 = 0: FOR D3 = 1 TO D:D2 =
     D2 +  INT (S *  RND (1) + 1)
     : NEXT :D2 = D2 - A * MD%(DF
     ,8): IF D2 ( 1 THEN  PRINT "
        BLOW BOUNCES OFF ARMOUR":
      PRINT : RETURN
7640 MD%(DF,13) = MD%(DF,13) + D2
     : IF MD%(DF,13) )  = MD%(DF,
     1) THEN 7700
7645 A = MD%(DF,14):FD%(A) = FD%(
     A) + D2
7649  REM
*** TELL PLAYER HOW HE
     ALTHY THE DEFENDER IS. ***
7650  PRINT : PRINT MN$(DF);: ON
      INT (MD%(DF,13) * 5 / MD%(D
     F,1) + 1) GOTO 7655,7660,766
     5,7670,7675
7655  PRINT " IS IN GOOD SHAPE.":
      RETURN
7660  PRINT " IS LIGHTLY INJURED.
     ": RETURN
7665  PRINT " IS BADLY INJURED.":
      RETURN
7670  PRINT " IS VERY BADLY INJUR
     ED.": RETURN
7675  PRINT " IS AT DEATH'S DOOR.
     ": PRINT "  KNOCKING LOUDLY
     .": RETURN
7700  FLASH : PRINT MN$(DF);" IS
     DEAD!": NORMAL
7702 A = MD%(DF,14):FD%(A) = FD%(
     A) + MD%(DF,1) - MD%(DF,13) +
     D2:MD%(DF,5) = 0
7703  REM
*** PUT THE DEAD BODY
     IN THE ROOM. ***
7704 A2 = DF + NZ - NM:AD%(A2,4) =
     ROOM
7705  REM
*** DROP WEAPONS, DISA
     RM, REDUCE COURAGE (IN CASE
     MONSTER GETS RESURRECTED).
7706  IF MD%(DF,9) ) 0 THEN AD%(M
     D%(DF,9),4) = ROOM:MD%(DF,10
     ) = MD%(DF,10) - AD%(MD%(DF,
     9),5):MD%(DF,9) =  - 1:MD%(D
     F,4) = MD%(DF,4) / 2
7709  REM
*** IF DEAD CHARACTER
     IS PLAYER CHARACTER, END GAM
     E ***
7710 A2 = DF + NZ - NM:AD%(A2,4) =
     ROOM: FOR AJ = 1 TO NA: IF A
     D%(AJ,4) =  - DF - 1 THEN AD
     %(AJ,4) = ROOM
7720  NEXT : IF MD%(DF,9) ) 0 THEN
     AD%(MD%(DF,9),4) = ROOM:MD%(
     DF,10) = MD%(DF,10) - AD%(MD
     %(DF,9),5):MD%(DF,9) =  - 1:
     MD%(DF,4) = MD%(DF,4) / 2
7740  IF DF = 0 THEN DIE = 1: GOTO
     2000
7900  PRINT : RETURN
8000  REM
***** FLEE COMMAND ***
     **

8010  IF  NOT NBTL THEN  PRINT : PRINT
     "THERE'S NOTHING TO FLEE FRO
     M!": PRINT : GOTO 100
8020  FOR D = 1 TO 6: IF S$ = C$(
     D) THEN V$ = S$:S$ = "FLEE":
      GOTO 3000
8030  NEXT
8399  REM
*** FIND A PLACE TO FL
     EE TO ***
8490 DR$ = "": GOSUB 8500: IF DR$
      = "000000" THEN  PRINT : PRINT
     "THERE'S NO PLACE TO GO!!": GOTO
     100
8495  GOTO 3500
8500 K1 = 0: FOR D = 1 TO 6:K = E
     D%(ROOM,D):DR$ = DR$ +  STR$
     (K): IF K ) 0 THEN K1 = K1 +
     1:DR%(K1) = K
8510  NEXT :RR =  RND (1) * K1 +
     1:R2 = DR%(RR)
8520  RETURN
9000  REM
***** GIVE COMMAND. **
     ***

9010 A2 = 1
9019  REM
*** SEE IF S$ CONTAIN
     S A MONSTER NAME ***
9020  FOR M = 1 TO NM: IF  LEFT$
     (S$, LEN (MN$(M))) = MN$(M) AND
     MD%(M,5) = ROOM THEN 9050
9029  REM
*** IF NOT, GET MONSTE
     R NAME ***
9030  NEXT : IF A2 = 1 THEN A2 =
     2: PRINT : INPUT "WHO TO GIV
     E TO? ";S$: GOTO 9020
9040  PRINT : PRINT "NOBODY HERE
     BY THAT NAME!": PRINT : GOTO
     100
9050 S$ =  MID$ (S$, LEN (MN$(M))
      + 1)
9060  IF  LEFT$ (S$,1) = " " THEN
     S$ =  MID$ (S$,2): GOTO 9060

9064  REM
*** IF VALUE OF S$ )0,
      S$ IS A NUMBER. ASSUME PLAY
     ER WANTS TO GIVE GOLD PIECES
      ***
9065  IF  VAL (S$) ) 0 THEN 9500
```

```
 9069  REM
*** SEE IF S$ CONTAINS
      AN ARTIFACT NAME ***
 9070  FOR A = 1 TO NA: IF  LEFT$
      (S$, LEN (AN$(A))) = AN$(A) AND
      AD%(A,4) =  - 1 THEN 9095
 9079  REM
*** IF NOT, GET AN ART
      IFACT NAME ***
 9080  NEXT : IF A2 = 2 THEN A2 =
      3: PRINT : INPUT "WHAT TO GI
      VE? ";S$: GOTO 9065
 9090  PRINT : PRINT "YOU AREN'T C
      ARRYING A ";S$;"!": PRINT : GOTO
      100
 9095  IF A = MD%(0,9) THEN MD%(0,
      9) =  - 1
 9100  PRINT : PRINT "OKAY.":AD%(A
      ,4) =  - M - 1:MD%(M,14) = 0
      :MD%(M,3) = MD%(M,3) * (1 +
      AD%(A,1) / 100)
 9110  IF MD%(M,10) <  > 0 AND MD%
      (M,9) =  - 1 THEN MD%(M,9) =
      A:MD%(M,10) = MD%(M,9) + AD%
      (A,5):MD%(M,4) = 2 * MD%(M,4
      )
 9120 R3 = ROOM: GOSUB 3600: GOTO
      5050
 9499  REM
*** GIVE GOLD ***
 9500  IF  VAL (S$) > GOLD THEN  PRINT
      : PRINT "YOU AREN'T CARRYING
      THAT MUCH GOLD OF       YOUR
      OWN!": PRINT : GOTO 300
 9510 GOLD =  INT (GOLD -  VAL (S$
      )):MD%(M,14) = 0:MD%(M,3) =
      MD%(M,3) * (1 +  VAL (S$) /
      100): GOTO 9120
10000  REM
***** INVENTORY COMMA
      ND *****

10010  PRINT : PRINT "YOU ARE CAR
      RYING THE FOLLOWING--":A0 =
      0: FOR A = 1 TO NA: IF AD%(A
      ,4) =  - 1 THEN  PRINT " ";A
      N$(A); LEFT$ (" (READY WEAPO
      N)",1 + 20 * (A = MD%(0,9)))
      :A0 = A0 + 1: IF A0 = 15 THEN
       PRINT : PRINT "(HIT ANY KEY
       TO CONTINUE)";: GET A$: PRINT
      :A0 = 0
10020  NEXT : GOTO 300
11000  REM
***** BLAST SPELL ***
      **

11010  GOSUB 4900
11469  REM
*** SEE IF TARGET IS
      BLASTABLE ***
11470  FOR M = 1 TO NM: IF S$ <  >
      MN$(M) OR MD%(M,5) <  > ROOM
       THEN  NEXT : PRINT : PRINT
      "YOU CAN'T BLAST ";S$;"!": PRINT
      : GOTO 100
11479  REM
*** ATTEMPT TO BLAST
      THE TARGET. SUC = SUCCESS FL
      AG ***
11480 S = 1: GOSUB 11500: IF  NOT
      SUC THEN 300
11489  REM
*** SUCCESSFUL BLAST.
      ***
11490  PRINT : PRINT "DIRECT HIT!
      ": PRINT :S = 6:D = 1:DF = M
      :A = 0: GOSUB 7635: PRINT : GOTO
      300

11499  REM
*** BLAST SUBROUTINE.
      SEE IF SPELL IS SUCCESSFUL
      ***
11500 SUC = 0: IF SA%(S) THEN RL =
      FN R(X):SUC = (((RL < S2%(S
      )) OR (RL < 6)) AND (RL < 96
      ))
11510  IF SUC THEN 11520
11511  REM
*** IF ROLL=1 THEN SP
      ELL HAS BEEN FORGOTTEN. ***
11512  IF  FN R(X) = 1 THEN  GOSUB
      11600: RETURN
11513  REM
*** FAILURE. ***
11514  PRINT "  NOTHING HAPPENS."
      RETURN
11519  REM
*** TEST FOR SKILL IN
      CREASE. ***
11520 RL =  INT (100 *  RND (1) +
      1): IF RL > S2%(S) THEN S2%(
      S) = S2%(S) + 2:SA%(S) = SA%
      (S) + 2
11530 S2%(S) = S2%(S) / 2
11540  RETURN
11599  REM
*** FORGET SPELL SUBR
      OUTINE. ***
11600  PRINT : PRINT "THE STRAIN
      OF ATTEMPTING TO CAST THIS
       SPELL OVERLOADS YOUR BRA
      IN AND YOU       FORGET IT CO
      MPLETELY.": PRINT :SA%(S) =
      0: RETURN
12000  REM
***** HEAL SPELL ****
      *

12010 S = 2: GOSUB 11500: IF  NOT
      SUC THEN 300
12020  PRINT : IF MD%(0,13) THEN
       PRINT "SOME OF YOUR WOUNDS
      SEEM TO CLEAR UP.": PRINT
12030 MD%(0,13) = MD%(0,13) -  INT
      (10 *  RND (1) + 1): IF MD%(
      0,13) < 0 THEN MD%(0,13) = 0

12040 DF = 0: GOSUB 7650: GOTO 30
      0
13000  REM
***** POWER SPELL --
      RE-WRITE FOR SPECIAL EFFECTS
       *****

13005  REM
*** RANDOM EFFECTS **
      *
13010 S = 4: GOSUB 11500: IF  NOT
      SUC THEN 300
13019  REM
*** RESURRECTION OF A
      DEAD MONSTER? ***
13020 RAISE = 0: FOR M = 1 TO NM:
      M2 = NZ - NM + M: IF AD%(M2,
      4) = ROOM OR AD%(M2,4) =  -
      1 THEN RAISE = 1: PRINT MN$(
      M);" COMES ALIVE!":AD%(M2,4)
       = 0:MD%(M,5) = ROOM:MD%(M,1
      3) = 0:WT = WT - AD%(M2,3)
13030  NEXT M: IF RAISE THEN R3 =
      ROOM: GOSUB 3600: GOTO 300
13039  REM
*** AN ARTIFACT VANIS
      HES? ***
```

```
13040  FOR A = 1 TO NA: IF AD%(A,
     4) = ROOM THEN RAISE = 1: PRINT
     AN$(A);" VANISHES!":AD%(A,4)
     = 0
13050  NEXT A: IF RAISE THEN  PRINT
     : GOTO 300
13059  REM
*** ROOF COLLAPSES? *
     **
13060 RR =  INT (100 *  RND (1) +
     1): IF RR < 11 THEN  PRINT :
     PRINT "THE SECTION OF TUNNE
     L YOU ARE IN ": PRINT "  COL
     LAPSES AND YOU DIE.":DIE = 1
     : GOTO 2000
13079  REM
*** SONIC BOOM? ***
13080  IF RR < 86 THEN  PRINT : PRINT
     "YOU HEAR A VERY LOUD SONIC
     BOOM": PRINT "  THAT ECHOES
     THROUGH THE TUNNELS.": PRINT
     : GOTO 300
13089  REM
*** HEALING? ***
13090  IF RR > 95 THEN  PRINT : PRINT
     "ALL OF YOUR HITS ARE HEALED
     .": PRINT :MD%(0,13) = 0: GOTO
     300
13100  GOTO 14020
14000  REM
***** SPEED SPELL: IN
     CREASE AGILITY AND CHANCE TO
     HIT *****

14010 S = 3: GOSUB 11500: IF  NOT
     SUC THEN 300
14020  IF  NOT SPD THEN MD%(0,10)
     = MD%(0,10) + 2 * MD%(0,2):
     MD%(0,2) = 2 * MD%(0,2)
14030 SPD = SPD +  INT (25 *  RND
     (1) + 10): PRINT : PRINT "YO
     U CAN FEEL THE NEW AGILITY F
     LOWING": PRINT "  THROUGH Y
     OU!": PRINT : GOTO 300
15000  REM
***** SMILE COMMAND:
     TRY TO MAKE FRIENDS *****

15010  PRINT : FOR M = 1 TO NM: IF
     MD%(M,5) <  > ROOM THEN 1510
     0
15020  ON MD%(M,14) GOTO 15030,15
     040,15050
15030  PRINT MN$(M);" GROWLS AT Y
     OU": GOTO 15100
15040  PRINT MN$(M);" IGNORES YOU
     ": GOTO 15100
15050  PRINT MN$(M);" ";V$;"S BAC
     K": GOTO 15100
15100  NEXT M: PRINT : GOTO 300
16000  REM
**** SAY COMMAND ****
     *

16010  GOSUB 4900
16030  IF S$ = "BLAST" THEN V$ =
     S$:S$ = "": GOTO 11000
16040  IF S$ = "HEAL" THEN 12000
16050  IF S$ = "POWER" THEN 13000

16060  IF S$ = "SPEED" THEN 14000

16900  PRINT : PRINT "OKAY, '";S$
     ;"'": PRINT : GOTO 300
17000  REM
***** READY A WEAPON.
     *****

17010  GOSUB 4900
17019  REM
*** IS HE CARRYING TH
     E WEAPON? ***
17020  FOR A = 1 TO NA: IF AN$(A)
     <  > S$ OR AD%(A,4) <  > -
     1 THEN  NEXT : PRINT : PRINT
     "YOU AREN'T CARRYING IT!": PRINT
     : GOTO 100
17029  REM
*** IS IT REALLY A WE
     APON? ***
17030  IF AD%(A,2) < 2 THEN  PRINT
     : PRINT "THAT ISN'T A WEAPON
     !": PRINT : GOTO 100
17979  REM
*** READY IT. ***
17980 MD%(0,10) = (EA + AE) * ( -
     EA > AE) + AD%(A,5) + WA%(AD
     %(A,6)) + 2 * MD%(0,2):MD%(0
     ,9) = A:MD%(0,11) = AD%(A,7)
     :MD%(0,12) = AD%(A,8)
17990  PRINT : PRINT "READIED.": PRINT
     : GOTO 300
18000  REM
*** SAVE GAME ***

18010  PRINT : PRINT "DO YOU WANT
     TO SAVE THIS GAME? ": PRINT
     "(Y/N) :";
18020  GET A$: IF A$ <  > "Y" AND
     A$ <  > "N" THEN 18020
18030  PRINT A$: IF A$ = "N" THEN
     100
18035 DK$ =  CHR$ (4)
18040  PRINT DK$;"CLOSE":X =  FRE
     (0)
18050  PRINT DK$"BSAVE GAME.PTRS,
     A$69,L8"
18060  PRINT DK$"BSAVE GAME.SVAR,
     A"; PEEK (105) +  PEEK (106)
      * 256;",L"; PEEK (109) +  PEEK
     (110) * 256 -  PEEK (105) -
      PEEK (106) * 256 + 1
18070  PRINT DK$"BSAVE GAME.STR,A
     "; PEEK (111) +  PEEK (112) *
     256;",L"; PEEK (115) +  PEEK
     (116) * 256 -  PEEK (111) -
      PEEK (112) * 256 + 1
18080  PRINT : PRINT "THE GAME IS
     NOW SAVED. BOOT ON THIS": PRINT
     "  DISKETTE TO RESTART THE G
     AME."
18090  END
19000  REM
*** RESTART OLD GAME *
     **

19010  PRINT : PRINT "DO YOU WANT
     TO RESTART THIS GAME?": PRINT
     "  Y OR N:";
19020  GET A$: IF A$ <  > "Y" AND
     A$ <  > "N" THEN 19020
19030  PRINT A$: IF A$ = "N" THEN
     END
19035 DK$ =  CHR$ (4)
19040  PRINT DK$"BLOAD GAME.PTRS"
     : PRINT DK$"BLOAD GAME.SVAR"
     : PRINT DK$"BLOAD GAME.STR"
```

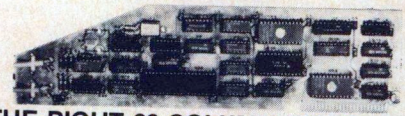
```
19050  PRINT DK$;"DELETE GAME.PTR
     S": PRINT DK$;"DELETE GAME.S
     VAR": PRINT DK$;"DELETE GAME
     .STR"
19060  PRINT DK$;"OPEN EAMON.DESC
  .  ,L256": PRINT DK$;"OPEN EAMO
     N.ROOMS,L64": PRINT DK$;"OPE
     N EAMON.ROOM NAMES,L64"
19070  GOTO 100
50000  REM
***** ERROR HANDLING
     ROUTINE *****
59000  REM
*** FIRST POKE IN A R
     OUTINE TO HELP DEFEAT APPLES
     OFT'S 'ONERR GOTO' BUGS. ***
59001  POKE 768,104: POKE 769,168
     : POKE 770,104: POKE 771,166
     : POKE 772,223: POKE 773,154
     : POKE 774,72: POKE 775,152:
     POKE 776,72: POKE 777,96: CALL
     768
59004 X =  FRE (0): REM ** THIS A
     LSO HELPS KILL BUGS.
59005  NORMAL
59006  REM
*** PRINT ERROR #, LINE # ***

59010 CODE =  PEEK (222):LN =  PEEK
     (218) + 256 *  PEEK (219)
59020  PRINT  CHR$ (7);"**** ERRO
     R #";CODE;" IN LINE ";LN
59030  IF CODE = 255 THEN  STOP :
     REM  ** BREAK ATTEMPTED
59199  PRINT
59200  PRINT "*** ATTEMPTING TO R
     ECOVER ***": PRINT : GOTO 10
     0
59999  END
60000  REM
*** THAT'S ALL, FOLKS! ***
```


219.95

FINALLY A RIGHT PRICE FOR THE RIGHT PRODUCT

micromax
presents
VIEWMAX-80



THE RIGHT 80-COLUMN VIDEO CARD FOR YOUR APPLE II*

It is more compatible with existing 80-column application software than other leading cards. And it is loaded with features. Check for yourself.

	RETAIL PRICE	7 X 9 CHAR DOT MATRIX	TRUE DESCENDERS	BUILT-IN SOFT VIDEO SWITCH	BUILT-IN INVERSE VIDEO CHAR.	FULLY PASCAL COMPATIBLE	SHIFT KEY SUPPORT
VIEWMAX-80	219	Yes	Yes	Yes	Yes	Yes	Yes
SMARTERM*	360	No	No	Yes	Yes	Yes	Yes
VIDEOTERM*	345	Yes	Yes	No	No	No	No
SUP'R TERM*	375	No	Yes	No	Yes	No	Yes
OMNIVISION*	295	No	No	No	Yes	No	Yes
VISION 80*	375	Yes	Yes	Yes	Yes	Yes	Yes
FULLVIEW 80*	379	Yes	Yes	Yes	Yes	Yes	Yes
WIZARD 80*	245	Yes	Yes	No	Yes	No	No

VIEWMAX-80 is functionally compatible with VIDEX's Videoterm and works easily with 80-column applications like CP/M, Pascal, WORDSTAR, AppleWriter II, Zardex, Easywriter, Letter Perfect, Magic Window, Apple PIE, VISICALC, ASCII EXPRESS, Transend, dBASE II, etc.

Shift Key Mod and Preboot disk for Visicalc and AppleWriter II available as low cost options. See your local Dealer or order direct. Visa/MasterCard accepted. Dealer & OEM inquiries welcome.

BACKED BY A FULL 2-YEAR WARRANTY.

DEALER & OEM INQUIRIES WELCOME.

micromax
6868 Nancy Ridge Drive
San Diego, CA 92121
(619) 457-3131

Also, Apple Parallel Printer Card; $79.95
Apple 40-Column Upper/Lowercase Adapter: $29.95
Z-80 CP/M Card: $149.95

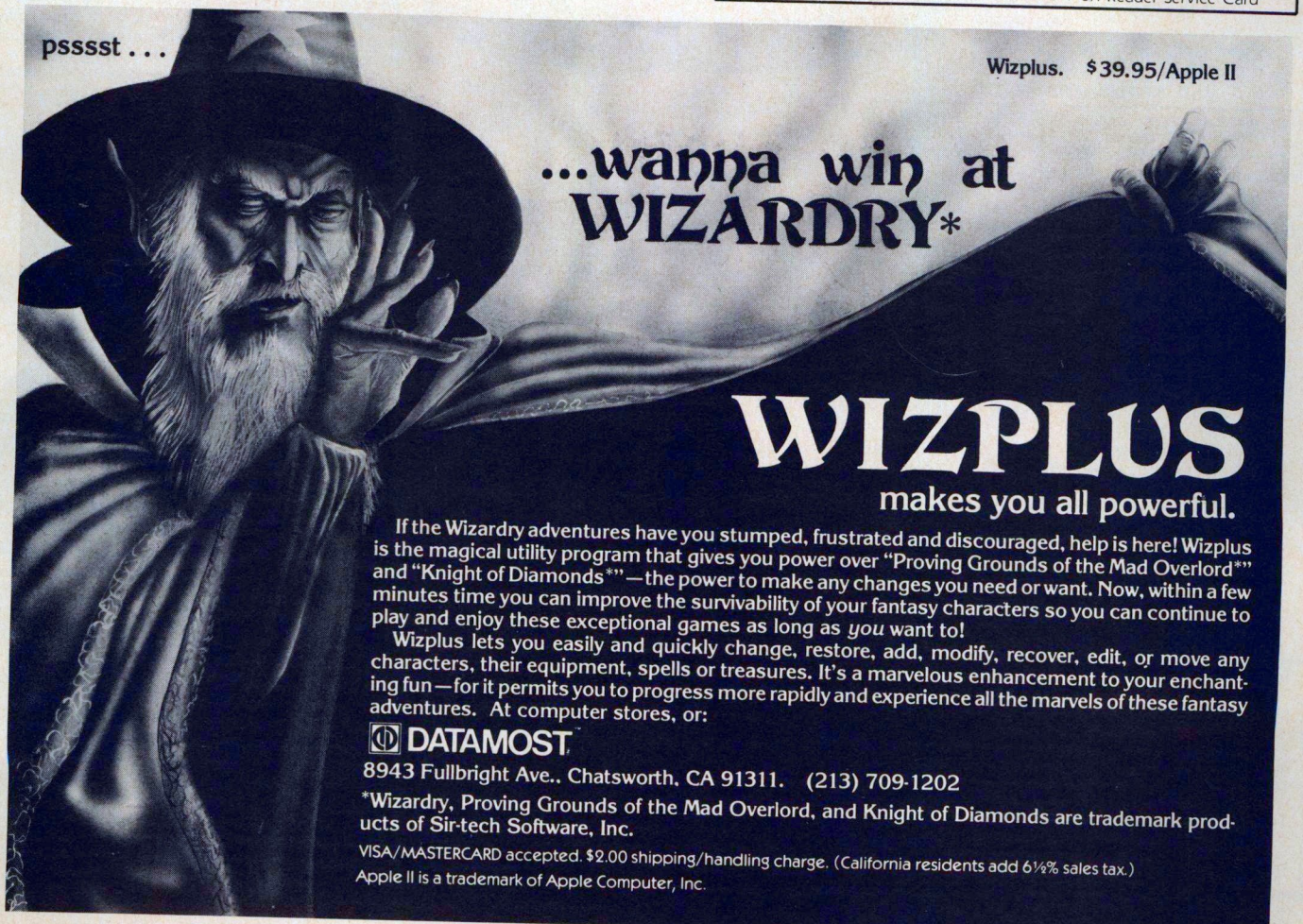
"WE MAXIMIZE YOUR MICRO INVESTMENT"

*Trademarks of Apple Inc, Advanced Logic Sys., Videx Corp, M&R Enterprises, Computer Stop, Vista Computers, Bit 3 Computers, Wesper Microsystems.

Circle 1 on Reader Service Card

psssst . . .

Wizplus. $39.95/Apple II



...wanna win at WIZARDRY*

WIZPLUS

makes you all powerful.

If the Wizardry adventures have you stumped, frustrated and discouraged, help is here! Wizplus is the magical utility program that gives you power over "Proving Grounds of the Mad Overlord*" and "Knight of Diamonds*"—the power to make any changes you need or want. Now, within a few minutes time you can improve the survivability of your fantasy characters so you can continue to play and enjoy these exceptional games as long as *you* want to!

Wizplus lets you easily and quickly change, restore, add, modify, recover, edit, or move any characters, their equipment, spells or treasures. It's a marvelous enhancement to your enchanting fun—for it permits you to progress more rapidly and experience all the marvels of these fantasy adventures. At computer stores, or:

DATAMOST

8943 Fullbright Ave., Chatsworth, CA 91311. (213) 709-1202

*Wizardry, Proving Grounds of the Mad Overlord, and Knight of Diamonds are trademark products of Sir-tech Software, Inc.

VISA/MASTERCARD accepted. $2.00 shipping/handling charge. (California residents add 6½% sales tax.)

Apple II is a trademark of Apple Computer, Inc.

Circle 2 on Reader Service Card

NEW Unlocked Apple® Utility Disks

Don't Blow Your Bucks on Locked-Up Uncopyable Apple Software.

NEW! Frame-Up

HI-SPEED GRAPHICS DISPLAY
BY TOM WEISHAAR

CREATE PROFESSIONAL PRESENTATIONS of intermixed hi-res, lo-res and text frames. Easy-to-use and FAST— hi-res images load in 2½-seconds! Paddles or Keyboard-advance frames in forward or reverse.

UNATTENDED SHOWS are possible with each frame individually pre-programmed to appear on the screen from 1 to 99 seconds.

TEXT SCREEN EDITOR lets you create your own b/w text "slides". Add type "live" from the keyboard during presentations if you want.

DISPLAY MODULE: Send entire presentations-on-disk to your friends and associates.

FRAME-UP: $29.50
(Includes Peeks/Pokes Chart)

Apple Mechanic

SHAPE-WRITER/BYTE-ZAP DISK
BY BERT KERSEY

SHAPE EDITOR: Keyboard-draw shapes for hi-res animation in your programs. Design proportionally-spaced typefaces with special characters. 6 fonts on the disk. Listable demos show how to use shape tables to animate games, graphics and professional Charts & Graphs.

BYTE-ZAP: Rewrite any byte on a disk for repair or alteration. Load entire sectors on the screen for inspection. Hex/Dec/Ascii displays and input. Complete instructions for making trick file names, restoring deleted files, etc.

MORE: Useful music, text and hi-res tricks for your programs. Educational documentation.

APPLE MECHANIC: $29.50
(Includes Peeks/Pokes Chart & Tip Book#5)

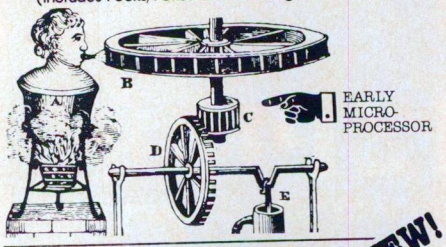
NEW! Typefaces

FOR APPLE MECHANIC

26 NEW FONTS for Apple Mechanic's Xtyper and Hi-Writer programs. Most are full 96-character fonts, large & small, of fully-editable characters. (Apple Mechanic required)

BEAGLE MENU: Use with your disks. Display only the filenames you want (e.g. only Applesoft files or only Locked files) for one-key cursor selection/execution. Space-on-disk, catalog scan, optional sector-number elimination.

TYPEFACES for Apple Mechanic: $20.00
(Includes Peeks/Pokes Chart & Beagle Menu Utility)



Beagle Bag NEW!

12-GAMES-PLUS ON ONE DISK
BY BERT KERSEY

TWELVE GREAT GAMES from the classic Beagle Bros collection— TextTrain, Slippery Digits, Wowzo, Magic Pack, Buzzword... Almost all of our "Game Pack" games, updated and re-released on one jam-packed, entertaining, unprotected disk.

COMPARE BEAGLE BAG with any one-game locked-up game disk on the market today. All 12 games are a blast, the price is right, the instructions are crystal clear, AND the disk is copyable. You can even change the programs or list them to LEARN, and see what makes them tick.

BEAGLE MENU TOO: See "Typefaces" above.

BEAGLE BAG: $29.50
(Includes Peeks/Pokes Chart & Beagle Menu Utility)

NEW! Flex Text

70-COLUMN TEXT UTILITY
BY MARK SIMONSEN

PRINT VARIABLE-WIDTH TEXT on the hi-res screens with normal Applesoft commands (including Htab 1-70). Normal, expanded & compressed text on same screen— no hardware!

ADD GRAPHICS TO TEXT or vice-versa. Run existing programs under Flex Text control. Easy to use and compatible with PLE© and GPLE.©

DOS TOOL KIT© FONT compatibility, or use Flex Text fonts. Select up to 9 fonts with ctrl-key commands. Print/List/Catalog in any style! Custom TEXT CHARACTER EDITOR included.

FLEX TEXT: $29.50
(Includes Peeks/Pokes Chart; requires monitor)

Utility City

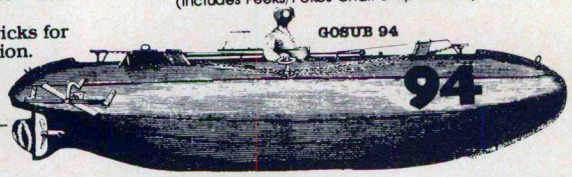
21 UTILITIES ON ONE DISK
BY BERT KERSEY

LIST FORMATTER prints each program statement on a new line. Loops indented with printer page breaks. A great de-bugger! Also...

MULTI-COLUMN catalogs for printouts, auto-post Run-number & Date in programs, put invisible commands in programs, create INVISIBLE file names, alphabetize/store info on disk, convert decimal to hex or INT to FP, renumber to 65535, append programs, dump text-screen to printer...

MORE TOO: 21 Programs Total, a best-seller!

UTILITY CITY: $29.50
(Includes Peeks/Pokes Chart & Tip Book#3)




```
10  FOR A = 1 TO 22: PRINT CHR$(ASC (MID$(
    "IJ—!IPX(T!ZPVS!TJTUFS@", A, 1))—A/A);
20  FOR B = 1 TO 4: C = PEEK(49200): NEXT B, A
```

DOS Boss

DISK COMMAND EDITOR
BY BERT KERSEY & JACK CASSIDY

RENAME COMMANDS & ERROR MESSAGES: "Catalog" can be "C"; "Syntax Error" can be "Oops" or anything you want. Protect your programs; unauthorized save-attempt can produce "Not Copyable" message. Also LIST-prevention and one-key program-run from catalog.

CUSTOMIZE DOS: Change Disk Volume heading to your message. Omit/alter catalog file codes. Fascinating documentation and tips; hours of juicy reading and Apple experiments.

ANYONE USING YOUR DISKS (booted or not) will be formatting DOS the way you designed it.

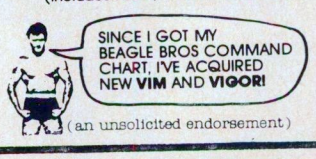
DOS BOSS: $24.00
(Includes Peeks/Pokes Chart & Tip Book#2)

Tip Disk#1

100 TIP BOOK TIPS ON DISK
BY BERT KERSEY

100 LISTABLE PROGRAMS from Beagle Bros Tip Books 1-4. Make your Apple do things its never done! All programs changeable for experimentation. Includes our Apple Command Chart: ALL Applesoft, Integer & DOS Commands!

TIP DISK#1: $20.00
(Includes Peeks/Pokes and Apple Command Charts)



(an unsolicited endorsement)

"APPLE" is a registered trade mark of You-Know-Who.

NEW! ProntoDOS

HIGH-SPEED DISK UTILITY
BY TOM WEISHAAR

HIGH-SPEED DOS! Take a look—

Function	Normal	Pronto
BLOAD HI-RES IMAGE	10 sec.	3 sec.
BSAVE HI-RES IMAGE	12 sec.	6 sec.
LOAD 60-SECTOR PROGRAM	16 sec.	4 sec.
SAVE 60-SECTOR PROGRAM	24 sec.	9 sec.
BLOAD LANGUAGE CARD	13 sec.	4 sec.
TEXT FILES	(no change)	

BOOT PRONTO-DOS or any updated normal-3.3 disk. Create new ProntoDos disks with the normal INIT command. ProntoDos is compatible with ALL DOS COMMANDS and performs normally with almost ALL programs, including CopyA.

MORE DISK SPACE: ProntoDos frees-up 15-extra-sectors per disk, almost one full track!

PRONTO-DOS: $29.50
(Includes Peeks/Pokes Chart)

Alpha Plot

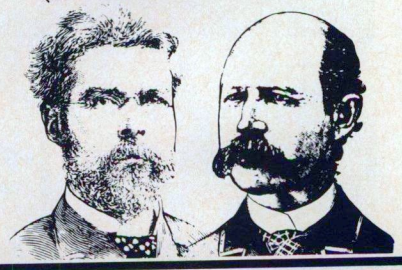
HI-RES GRAPHICS/TEXT UTILITY
BY BERT KERSEY & JACK CASSIDY

DRAW IN HI-RES, on 2 pages, using keyboard or paddles/joystick. See lines before plotting. Mixed-colors and reverse (background opposite). Fast circles, boxes and ellipses; filled or outlined.

COMPRESS HI-RES PIX to 1/3 Disk-Space. Superimpose pages or re-locate any rectangular image area anywhere on either hi-res page.

HI-RES TEXT: Proportional spacing, adjustable character size and color, upper/lower case, no tab limits, sideways typing for graphs.

ALPHA PLOT: $39.50
(Includes Peeks/Pokes Chart & Tip Book#4)



Beagle Bros™ MICRO SOFTWARE

Where to Buy Beagle Bros Disks:

MOST APPLE DEALERS carry Beagle Bros software. If yours doesn't, get on his case. Or order directly from us for IMMEDIATE SHIPMENT—

Visa/MasterCard/COD, call TOLL FREE:
Nationwide: 1-800-854-2003 ext. 827
California: 1-800-522-1500 ext. 827
Alaska/Hawaii: 1-800-854-2622 ext. 827

OR mail U.S.check, money-order or Visa/MC #'s to BEAGLE BROS, Dept. Z
4315 SIERRA VISTA / SAN DIEGO, CA 92103

Please add $1.50 First Class shipping, any size order. Overseas add $4.00. COD add $3.00. California add 6% ALL ORDERS SHIPPED IMMEDIATELY.

A Call —A.P.P.L.E. REVIEW:

Ampersand Utilities

David Sparks/Call —A.P.P.L.E. Staff Writer

THIS review is dedicated to every Apple user who ever said, "If only Applesoft had a command to do...(you name it)."

In Gilbert & Sullivan's "The Mikado," there is a character called The Lord High Everything Else. He gets all the jobs nobody else wants. It turns out that Applesoft has a Lord High Everything Else, too. He is the ampersand, "&." And, as Shakespeare put it, "thereby hangs a tale."

Call -A.P.P.L.E first disclosed the existence of the ampersand command in the autumn of 1978, back when the language we all know today was called Applesoft II, to distinguish it from a mythical earlier version that came on cassette tape (!). Here are the words by which the world learned of this useful command:

& NOW, the AMPERSAND

Inside the elaborate labyrinths of Applesoft]l there lurks a mysterious and shady character. This character can perform marvelous and incredible feats however, when beckoned forth to perform his many functions. This character goes by the name of Mr. Ampersand. We'll call him & for short. By cleverly using Mr. &, the user can extend Applesoft]l indefinitely. How?

When Applesoft is executing a program, if it encounters an & at the beginning of a statement, a JSR to location $3F5 is executed. At this time, the user is free to do anything from assembly language...

This user, for one, was left high and dry, not being skilled at assembly language. Much later, in May of 1980, David Lingwood described how to get started using the ampersand. He called it "The best kept secret in all of Applesoft." And he concluded, "Mr. Ampersand can be a helpful character; I just wish the manual writers had seen fit to introduce him properly."

Initially, a lot of folks applied the ampersand as a simple way of calling just one machine language subroutine. Program listings appeared in which, after properly setting things up, a line such as "235 &" would use the ampersand all alone. No clue given as to what it meant, you just had to know or take it on faith.

To be sure, we had used the ampersand before. The Renumber program on the System Master uses it, as does CRAE from Highlands Computer Services, and other editing utilities. Apple Computer, Inc. marketed a package of Applesoft extensions once, though no one in this writer's acquaintance owns a copy.

Came 1981. *Call -A.P.P.L.E. In Depth #1* contained a reprint of John Crossley's "Applesoft Internal Entry Points," from Apple Orchard. Also included: "Print Using & Friends," the first published *Package* of routines this writer had seen. Craig Peterson described quite well how to add on to Applesoft through the ampersand. The reader is referred to those articles for the particulars.

Applesoft extensions: additional, desirable commands that make Applesoft programs do more things better with less effort by the programmer. Without needing to become an expert on the mysteries of machine language, the programmer gains powerful new tools which are easy to use and which strongly resemble friendly BASIC. Some, like PRINT USING, are actually found in many other BASIC versions, and make their way to Applesoft through the good offices of the ampersand. Others satisfy that "if only" craving, replacing many lines of Applesoft with a single program statement, as in the SORT command, and executing such frequently-used routines at the near-light speed of machine language. You get the power and the glory; somebody else has already done the work.

As 1982 drew to a close, a number of such packages began to appear from commercial vendors. (Come to think of it, what other kind of vendors are there?) We have reviewed as many as we could obtain. This article will cover each one in turn, and conclude with some thoughts to assist the reader in choosing one for purchase.

Our review covers: "Amperware," Bob Hogan/Scientific Software Products, Inc.; "The Routine Machine," Peter Meyer with Craig Peterson, Glen Bredon, and Roger Wagner, Southwestern Data Systems; "Amper-Magic," Bob Nacon, Anthro-Digital Inc.; "Amper Manager," R. Sutcliffe, presently published by Mr. Sutcliffe and soon to be available to *A.P.P.L.E.* members; and "Amper-Soft," Cornelis Bongers, MicroSparc, Inc.

Two things the reader should know up front. 1) There will be no "pick of the crop." These packages are all well-done and each has strengths which make it most appropriate for particular applications. Our emphasis will be on helping the reader's own needs guide his choice intelligently. 2) This writer was, late in December, 1982, negotiating with Southwestern Data Systems with respect to selling them a program we co-authored. We elected to go ahead and include their package in the review, with disclosure of our conflict of interest. As Thomas Jefferson wrote in a famous document, "Let the facts be submitted to a candid world." We hope we succeed in remaining objective.

There is an old saying that "Any critic can kill a play, but very few can write one." We were impressed with

—●→

all of the packages we reviewed, and have no intentions of "programmicide."

Every effort has been made to obtain the comments of the authors and vendors in advance of publication, and to incorporate their comments in the original review. A copy of this review went to each vendor in the first week of January, and they were urged to contact us with any amplifications, changes, or corrections. We believe that fairness to the reader demands fairness to the creators of what we review.

The packages have surprisingly few extensions in common. First in importance, no doubt, is a "PRINT USING" command. Applesoft's most glaring lack is this straightforward print formatting tool. It solves decimal point alignment problems, right-hand-side formatting problems, the proper insertion of commas in numbers, arranging numbers in nice columns, and much more. We heard of people who chose another computer instead of the Apple][because they wanted this command. Well, be assured that no matter which of the reviewed packages the user selects, he or she will obtain a flexible, powerful "PRINT USING" command. In fact, be pleased.

Four of the packages provide GOTO [aexpr] and GOSUB [aexpr] where [aexpr] is a numeric expression. Example: INPUT "CHOOSE YOUR RIDDLE: ";N : GOSUB N*200. The same four provide a means to INPUT colons, commas, and just about anything.

No other extension was found in each and all utility packages. Quite a number were present in more than one. Three had utilities to locate a substring within a string, or within a string array. A different three offered decimal to hexadecimal conversion routines, though only two of these provided hex back to decimal operations. Three had text screen positioning features along the lines of "PRINT AT X,Y;"[string]".

To see what the various authors felt worthwhile, consult the table which accompanies this article.

The programs are covered, one at a time, in accordance with the following outline:

I. Compatibility with PLE, CRAE, and GPLE, three commonly-used editing utilities
II. Compatibility with 80-column cards, where the documentation specifically mentioned it. We do not have one in our system, thus did not test this item.
III. Location of the package at runtime of an Applesoft program, and how much memory it consumes.
IV. User-supplied machine-code modules, the ease of including them, special constraints on their use, if any, and amount of help the documentation affords the user in this effort.
V. The documentation itself, reviewed for accuracy, completeness, thoroughness, and every other desirable adjective we could think of.
VI. Policy of the publishers regarding backups, and use in commercial programs written by users of the packages.
VII. Syntactical conventions, that is, how much like Applesoft are the extended commands, are they readily understandable from the context of a program statement in which they might appear, and how easily could someone else follow a program written by the user? Oh, you say you don't want anybody trailing you through your program listings? Read on...
VIII. Minimum system requirements, where stated in the documentation. If we could not find such a statement, we will say so. If the vendor lets us know, we will pass that along, too.
IX. Special features worthy of note.

An old joke is that there are two kinds of people in the world: those who divide the world into two kinds of people, and those who don't.

Well, at the risk of a cliche, there are two ways to implement extensions to Applesoft. Both involve the use of command modules written in machine language. One way is to somehow attach the modules to the program itself. The other is to place them more or less permanently in some part of the Apple][memory so that programs can call upon them.

The first method allows the programmer to select and attach to his or her works only those modules which carry out the extensions he or she wants to use. Memory isn't tied up with a block of machine code serving no purpose. What is more, the programs can be saved with their attendant extensions, and can be loaded and run normally, without any other steps being required at "run time." But the extensions won't be available to any other program, and the process of linking them to a program does involve extra steps for the programmer.

Two of the packages we reviewed, "Amper-Magic" and "The Routine Machine," follow the first plan. They include on disk large libraries of dandy commands, plus a quickly learned, easily understood and flexible program to attach the desired commands to Applesoft programs. The user is able to give the additional commands his own names, making them very personal enhancements indeed. Should the programmer forget what "& BAMBOOZLE,A$, X" means, the linking program can dig out the name of the disk file it's related to. The two go to varying lengths to encourage the user to add command modules of his or her own writing, or to type them in from magazines.

The second method requires no special linkage procedures, makes available to all programs a consistently named set of extra commands, and is easy to use. But it ties up a large amount of memory when sometimes only one or two extensions are desired. As we will see, one of the packages reviewed has turned this all-or-none requirement to its advantage. A second consideration is that the package of extensions has to be available to the system and loaded in at or prior to run time.

Our remaining three packages are of this second kind. Load them once, say, when booting, and every program you write can immediately make use of every routine they contain. In fact, all three include commands that are quite useful from immediate mode, becoming tools in the hand of the pro-

grammer during development, as well as levers multiplying the power of the programs themselves. In most cases, the documentation describes how user-supplied command modules can be "hooked-in" as well, though not as neatly nor flexibly as with the first method. See our concluding remarks for some thoughts on having the best of both worlds.

Amperware

Scientific Software Products Inc.
3171 Donald Ave.
Indianapolis, IN 46224
$49.95

This is one of the all-or-nothing packages, the second type described above. It is an excellent utility for programmers who must turn out a lot of file-handling programs with sophisticated input, printer output, and sorting requirements. The INPUT extension is perhaps the most impressive single extension we encountered. With only a few minutes of learning how to use it, the programmer can turn out tidy, sparse code to lay out a first-class input form on the screen, and make it almost impossible for the user to make a data-entry mistake. Absolutely ideal for a customized database application.

The user can specify the acceptable length of an input, whether it is limited to numeric data only, where on the screen the input is to be taken, enter upper and lower case, specify defaults to display and use, and declare how to handle any special characters during input (use the return key to tab around the form for example, then [Ctrl-C] to accept all data on the screen). All with almost Spartan simplicity in the Applesoft program itself.

A glance at the comparisons table shows some unique features for users of 80-column cards: a modified HOME to clear the whole screen, a WINDOW command that works in the 80 X 24 environment, and an PRINT command to select outputting upper and lower case together for those of us lacking lower case display.

It is one of two packages to offer binary saves and loads of arrays to and from disk. This is about 16 times faster than normal DOS READ and WRITE procedures. There is a SRT (sort) command which can do multi-key indexed sorts of any type array, as well as regular sorts in which the data are physically exchanged. There is a FIND command to locate a substring within a larger string, or within an array, or both at once. There is a CLEAR command to erase an array so that it can be re-DIMmed in a running program. Want to be sure your growing file will fit on the disk (so you don't die of DISK FULL ERROR after four hours of data entry)? There is a SPACE command to allow the program to test disk space.

Nice. What about the squawks? The package must be in memory to be used. It lives in memory between $84A0 and $9CFF. So do lots of other useful Applesoft helpers, such as PLE and CRAE. There is a low-memory version on disk, named Underware and running from $803 on up, but I could not get it and PLE to cohabit my Apple peacefully. Either they would hang the system, or die a beeping, asterisked death in the monitor. I am pleased to report that there appear to be no conflicts with the new GPLE.LC properly installed in the language card.

When we called Scientific Software for help on this, their man Mark Overhage got back to us the same day. Quick and courteous, they. Subsequent conversations with program author Bob Hogan found him busily solving the problem so that future shipments will peacefully co-exist with PLE in the lower 48.

The package occupies 6,239 bytes whether you use all of it or not. To add any routines supplied by the user, the module must be BLOADed, and the starting address has to be POKEd into special places inside Amperware. This was documented in an addendum, a loose sheet of paper in the back of the manual.

The manual is done in very tiny type. It is quite well laid out, though. A proper Introduction, a summary of the basic techniques to get the user going quickly, a chapter on Advanced Techniques to further amaze, and a (believe it) Complete Reference. In that order. It's the only manual in the bunch to provide an index. Now where's that magnifying glass? (In fairness it must be added that Dr. Hogan, being himself of normal eyesight, found the typesetting job equally distressing. While there is no truth to the rumor that he dismantled the printer in front of a physiology class he teaches at Indiana University, I can report that a new manual is being readied, in a more readable type size.)

The disk is copyable by any normal copy program. A legitimate purchaser receives permission to reproduce the program on his own disks. The manual describes the few, simple steps required to include the machine code in programs the user offers for sale.

The commands are called with clear words that signify what they do, and because the package is fixed, those command names will be the same everywhere Amperware goes.

Amper-Magic

Anthro Digital Inc.
P.O. Box 1385
Pittsfield, MA 01202
$75.00 Program Disk with utilities
$35.00 Utilities volume 2, without program

This is one of the mix-and-match packages. The disk comes with a utility for tacking machine code modules onto the end of an Applesoft program, plus lots of modules ready to run.

It seems to have no problems of compatability with any of the program editing tools this writer owns, PLE, CRAE, and GPLE.

With a program in memory, the user types EXEC AMPER-MAGIC and follows the self-prompting, menu-driven process from there. Here is one, slightly inelegant aspect: the user has to tell AmperMagic how much space to reserve for adding modules. This must be done each time the utility is EXECed. The manual explains that memory is measured by "pages," each 256 bytes long. The user needs to add up the lengths of the desired command modules' code, and divide it by 256 to get the answer to Amper-Magic's first question.

The instructions are very specific that exiting the utility must be done via menu option 8, "EXIT," else all will be lost. We ask, on behalf of those who are in the (bad?) habit of mashing the RESET key to bail out of things, could not the RESET vector have been redirected to the exit routine? That's quibbling, however, like complaining about the wrapper on a tasty candy bar.

We could find no mention of 80-column cards in the manual. Michael Heckman, President of Anthro-Digital, wrote us that no two of the 80-column cards on the market work exactly alike, so they elected not to specifically support any of them. Routines which don't do screen formatting, he suggested, shouldn't be affected by 80-column cards or the absence thereof.

AMPERSAND UTILITIES

Only 140 bytes of Amper-Magic code are needed at runtime, plus the actual command modules themselves. These elements are made one with the Applesoft program itself.

Clearly there is more to Amper-Magic than 140 bytes. Those are what we call the "front-end" code, the part that interfaces the modules to the program. The utility which accomplishes the attachment runs in memory above the Applesoft program being edited. It is wiped away by menu function 8, "EXIT," and does not need to be present when the program is actually run. This wiping away is why Amper-Magic needs to be EXECed each time it is needed.

Modules supplied by the more advanced user are just more entries in the library, as far as this type of package is concerned. Amper-Magic has an intriguing facility allowing object code to be entered at the keyboard. In the first draft of this article, we faulted the publishers for failing to document the condition of TXTPTR and the accumulator at the point where processing is passed to a module. We had a few moments of puzzlement deducing that the accumulator points to the byte just evaluated, and TXTPTR points to the next one in the program text. A JSR $B7 will load the accumulator from TXTPTR. Mr. Heckman said they would include this information in the manual henceforth.

The manual has a good tutorial, and an explanation of the hows and whys of the tool. The modules are described succinctly, each on its own page. While the pages of the tutorial and advanced technical information are numbered, the module reference pages themselves are not. Instead they are arranged alphabetically. We suppose the idea is to facilitate inclusion of reference pages from future "Command Library" disks the user might purchase. For our part, we missed the page numbers. Fair enough, Mr. Heckman?

We had a chance to evaluate the first of the add-on disks. It contains 27 new commands, and features what its distributors modestly describe as "the most powerful PRINT USING command *ever* for the Apple." The first disk contains a serviceable P-U, too. Volume two could be backed up normally. Volume one, with the attachment utility on board, could not be read by the COPYA program on our System Master.

Anthro-Digital says the user is free to sell programs containing Amper-Magic modules, provided proper credit is given. Best advice here is to contact them for precise wording. They say there is no charge for such commercial use. They also say that for an extra $15, they will put the user on a special mailing list to receive first notice of any bugs or new developments.

Minimum system requirements are clearly stated: 32K or more of RAM, Applesoft in ROM or RAM, DOS 3.3 and at least one disk drive. That's what it takes to run the attachment utility. Depending upon the modules being used, the minimum requirements to actually run a program over which the Amper-Magic wand has been passed could be considerably smaller.

Our conclusion is this is a capable utility with a complement of useful, general-purpose magic on Volume 1, and a taste of sorcery on Volume 2. We would keep it among our oft-used progamming aids, close by the computer.

Routine Machine

Southwestern Data Systems, Inc
P.O. Box 582
Santee, CA 92071
$64.95

This is the other cut-to-fit kind of ampersand utility. Like Amper-Magic, it offers a disk full of machine code extensions plus a utility program to attach the chosen ones to an Applesoft program. The program, thus enhanced, can be saved with its appended machine code and later run without the need for the Routine Machine utility to be in memory.

Comparing Routine Machine with Amper-Magic is something like comparing Cadillacs and Lincoln Continentals. Both give you a lot, but one has this itty bitty extra and the other has that teeny tiny goodie. The reader will recall that we want above all to be fair and impartial. But we will not carry that to the extreme of becoming an uninformative, plain brown wrapper. For instance, Routine Machine costs enough less to cover the price of two backup diskettes.

Routine Machine is compatible with PLE, CRAE, and GPLE also. One thing we did discover when renumbering with CRAE was that the appended code had best be saved as a block before the renumber, then re-linked to the program. This is also true of Amper-Magic's appended code, and is true of both when using Apple's RENUMBER program.

Again, no mention of 80-column cards in the manual.

The "front-end" section is 203 bytes long. Routine Machine's attachment utility occupies the space from $803 to $24FF and stays there until the user gets rid of it. It relocates the Applesoft program above itself and can be re-activated by CALL 2051. Of course, this means that now both the utility and the program itself occupy Hi-Res page 1. Programs will edit and run normally, but a program which issues the HGR command will immediately deny all knowledge of its own existence, and will vanish from the earth. This is not as bad as it sounds. It's good practice to SAVE a program before test-running it. And the Routine Machine comes with a short program you can BRUN to restore your Apple to normal operating conditions.

At the small cost of a temporary Hi-Res page conflict during program editing, this allows the user to keep a valuable "extra" close at hand. In addition to attaching machine code extensions, Routine Machine is a handy memory tool. A user can examine Applesoft lines as the Apple stores them in memory. Other features include a cross-reference of each ampersand command, and the line numbers in which they appear. These features are unique to Routine Machine, among the packages we review here.

User-supplied modules are easy to install, but they must already be present on disk as a binary file. And they must reside on a "legal" Routine Machine disk. That's OK, really, because each Routine Machine disk contains a copy program to back itself up three times. Each of those backups has well over 200 free sectors. If 200+ sectors times four disks isn't enough room, use FID to move modules from "non-legal" disks of your own onto "legal" disks.

Roger Wagner explained to me that this is a compromise to protect authors of modules on future library disks. The Routine Machine checks to be sure it is loading from a valid disk. Southwestern Data's unique approach to copy protection, by building in the ability to make three backups and three only, has been

called quite fair and reasonable by this magazine in past editorials. Maybe so. We found it a minor but pesky inconvenience.

Routine Machine is, on the whole, somewhat smoother in the tasks of attaching modules to a program. But the differences are more of style than of substance. Another person might find Amper-Magic's way of doing things more to his or her liking.

The manual is clear, well written, and gets the user going quickly. Advanced technical information is in a section separate from the fundamental, start-up stuff. The manual contains a number of reprints from Roger Wagner's "Assembly Lines" articles in Softalk Magazine, Describing the ins and outs of interfacing machine code to Applesoft. This for the user who wants to know more. Such knowledge isn't needed to use the package.

A table of contents including all the module descriptions is a big plus for this package. Alas, there is not an index. And again, the manual does not describe the condition of TXTPTR and the accumulator when the module takes over. In this case, the accumulator contains the byte to which TXTPTR points, a subtle difference from Amper-Magic and the cause of more head-scratching.

Information regarding use in commercially distributed programs was not given in the manual, but the user was instructed to contact Southwestern Data for such information and licensing. Wagner told us on the phone that they don't charge a fee, but do have some specific, simple requests to make regarding recognition and so forth. But don't take our word for it. Call them.

The user can give Routine Machine modules any name at all, of virtually any length. The names are put within quote marks, so Applesoft won't change any words to tokens. The manual makes all this perfectly clear. Plus, the back side of the program disk contains demonstration programs, one for each command in the library.

Again, no indication is given of minimum system requirements. Everything runs at the low end of memory. The programs, once the modules are attached, can run without Routine Machine being even in the same county, let alone the computer. Applesoft in ROM or on Language Card would be our guess at runtime. Add DOS 3.3 for using the attachment utility. We talked this over with the Southwestern Data folks, and they thought that sounds about right.

Some features unique to Routine Machine make it perhaps more attractive for game writers, fun havers, and graphics buffs. A sound effects command, a way to attach multiple shape tables, a built-in Hi-Res ASCII character set, plus (get this) Turtlegraphics. Yes, you can enjoy Turtlegraphics in Applesoft.

For the more serious side: RESTORE not just to a line number, but to a specific DATA element; a BLOAD that is four times faster than DOS; error-processing routines including one to print error messages without halting program operation.

Of course, PRINT USING and the GOTO and GOSUB to variables are included, as with all of these Ampersand packages. And subsequent libraries on disk with various specialized features are planned.

Amper Manager

R. Sutcliffe

Available soon from A.P.P.L.E.

This is the package for a would-be DOS wizard. It makes us think of something Caesar caused to be written on one of his campaigns, "The bridge built, Caesar crossed it." No mention of how the bridge was built or at what difficulty. Caesar did not bother himself with the building.

The DOS-crossing commands available when Amper Manager is on board allow astounding things to be done with few Applesoft lines. Imagine a single command that will read a sector from disk and dump it to the screen. We've seen listings many lines long to do the same. Even we could crank out a demo for our acquaintances with such applomb, as if to say, "The program written, Caesar blew them away."

We tested version 1.3. Version 1.4 will be the first to reach the buying public and will contain several enhancements. At this writing, all we could say about publishing plans was that a well-known, Apple-oriented organization with a reputation for powerful software at low prices was considering Amper Manager.

Amper Manager is specifically designed to function with PLE and CRAE. Hookups for both are programmed into the code. GPLE also worked just fine with it. Again, there was no mention of 80-column cards in the manual. Indeed, author Sutcliffe told us that he had not tested the program on an 80-column card. He also said that, as he uses standard Apple output protocols, he could think of no reason why an 80-column card would have any problems with his brainchild.

Three versions come on the disk. One runs at the top of memory in a 48-K machine, the second at $803, with programs above it, and the third is an R-type file about which we know nothing whatsoever. Depending on the version you use, it occupies from 5119 to 6066 bytes of memory, and must all be loaded to use any part.

Hooking in user-supplied utilities is not as easy as with Amper-Magic or Routine Machine. But the commands & 1 through & 9 jump to addresses of routines previously BLOADed and identified by the user with those numbers. The manual explains.

It wouldn't really be fair to comment on the manual, since the copy we had was such an early, developmental version. Let us quietly express the hope that substantial improvement occurs.

Again, because this was a developmental edition, we can't be sure of the backup policy. Our disk was copyable. The manual prohibited making more than two copies, period. Commercial use wasn't mentioned at all.

The commands in our version tended to be dense, not words but letters and as few of those as possible. If you've ever wanted to get a CATALOG with just a C, you may like this feature. For us, trying to remember what those letters actually mean is asking a lot of our poor, dim brains.

The manual we received doesn't say what the minimum system requirements might be. Mr. Sutcliffe, calling on his own nickel from British Columbia to Iowa, told us Amper Manager required at a minimum a 48K Apple with Applesoft in ROM or on the Language Card, and DOS 3.3 in its normal location, that is, where the System Master puts it when booted normally. Fans of fancy DOS-moving aerobatics take note. We straight-and-level types needn't be concerned.

What can it do? We've already hinted at strong disk editing features. Add, for openers, a utility to check a disk for bad sectors. Any such "craters" found are identified to the user, and if he or she desires, are

—●→

relegated to an "S"-type file called, you guessed it, "BAD SECTORS." That keeps them off limits when DOS goes looking for places to put the user's valuable data, precious programs, or, ahem, deathless prose.

The more sophisticated user will enjoy commands like &FMTDSK ON TRK A TO TRK B. Effectively, that will INIT only a part of a disk. It can also lead your disk through the valley of the shadow of death.

Three & PUT commands allow the user to write a modified DOS to disk, an entirely fresh CATALOG track, or just a new VTOC. If you don't grasp the significance of that, don't worry. The Sun will go right on shining upon you anyway.

Aside from the disk dazzlers, there are some interesting string handling commands. We admit that we had to ask Mr. Sutcliffe to tell us why we needed them. He made it abundantly clear. Taking a page from some other dialects of BASIC, Amper Manager allows the programmer to change specific characters within a string without going through Applesoft's sometimes contorted string functions. For instance, to change the value of A$ from "THIS IS OLD STUFF" to "THIS IS NEW STUFF," simply program & LET MID$(A$,9,3) = "NEW." Zounds!

The extension package alone is quite enough. But with it also come three or four Applesoft programs to demonstrate the features. They are far more than mere demos, however. One, a versatile EXEC file writer, should take the mystery out of that useful DOS device. Another, called The Director, is at minimum one of the most remarkable CATALOG management programs we have ever used. A third, The DOS Operator, can be used to modify DOS in many ways.

Our opinion is that programmers wanting to perform sophisticated disk management in their programs, and especially those without a powerful disk utility at present, would find this one to be the most appropriate of the packages under review.

AmperSoft
MicroSparc, Inc.
P.O. Box 639
Lincoln, MA 01773

That's "Sparc," with a "C," no relation to your reviewer. This is Cornelis Bongers' roundup of useful programs he has given us over the years in various magazine articles. For example, the program locates itself and DOS on the Language Card. So what? DOS takes up about 10K of memory. Adding a big wad of extensions absorbs even more, leaving less room for programs and data. But move DOS and the extensions to the Card, and instead of shrinking, the memory available to the program actually increases by about 10,000 bytes.

Obviously, such a plan makes a 16K RAM card a requirement. Therefore, the minimum system is a 64K Apple with Applesoft in ROM and DOS 3.3. The AmperSoft disk becomes the bootup disk. As you can see, if you have the Card, your system has extra clout.

The manual is excellent, as is everything Bongers writes. His emphasis is entirely on teaching the use of the enhancements. Very little of the operating theory is given. The manual is well-illustrated, brief, and to the point.

No provision is made for interfacing user-supplied Ampersand extensions. They have to be POKEd or BLOADed in the old way, and the Ampersand vector at $3F5 has to be re-directed by the user. It is also necessary for the user to re-connect AmperSoft by POKEs or from the Monitor. This makes AmperSoft somewhat less suitable for inexperienced programmers. Personal opinion.

80-column cards do not receive any special mention in the manual. David Szetela of MicroSparc did mention to us that the PRINT USING command works properly on the Videx card.

Math and science types will find this to be perhaps the most useful of the programs under review. Bongers has given us a powerful set of matrix commands: identity, transpose, inverse, minima, maxima, absolute maxima, and matrix multiplication. Like Amperware, he offers superfast disk storage and recall of numeric arrays, though not of string arrays. Regular as well as indexed sorts can be done on all three types of arrays. These are things which can be done easily enough in Applesoft, but the machine code is so much faster.

The command names are short, even non-existent in the case of the some matrix operations. Clearly an effort to simplify coding of the routines, it should not be a problem if this package is the only use made of the Ampersand in programs running under AmperSoft. Still, another programmer, not aware of AmperSoft syntax, would be unable to effectively read a listing of such a program.

The CATALOG command displays a special, unique character in front of each file name, which you may use to access the file. The one-finger, hunt-and-peck crowd will appreciate that.

Remembering that the Ampersand is a way to add needed features to Applesoft, Bongers gives us a faster garbage collection function. It is called with & FRE(0). An added twist is the ability to command & FRE(N) where N*1000 is the memory "pad" desired. & FRE(4) would call the garbage routine if less than 4000 bytes of free memory remained.

Not on the disk we reviewed, but added since according to Mr. Szetela, is a command to delete an array within a running program. That would be useful for an application dealing with a sequence of matrices of varying sizes and dimensions.

Recognizing that the relocated DOS represents a radical change in the operating system of the Apple, the AmperSoft disk includes updates to FID, MUFFIN, RENUMBER, COPYA, and PLE. GPLE on the Language Card shouldn't be used, because DOS and AmperSoft fully occupy it already.

Other utility programs or user-supplied extensions which expect to find DOS where it usually is, will not function properly.

Now to conclude. Applesoft is alright, but has some shortcomings. This magazine publishes a lot of articles pointing them out, and proposing solutions. Quite a number of

those articles present machine code extensions to Applesoft to improve or supplement it. Other articles show ways to greatly speed up certain processes with machine code. Finally, we have learned how to use many PEEKs, POKEs, and CALLs to perform magic.

The problem facing any Applesoft programmer, and especially those just getting started, is that such listings violate the "English-like" syntax of Applesoft. HOME we understand; CALL -936 leaves us in the dark, though it's the same thing. CALL 768 could mean anything, depending on what's been stored there.

And that's another thing — so many of the listings in the magazine articles are written to run at the same address. It takes some sophistication and experience to be able to understand, let alone resolve the memory conflicts. Certain lengthy listings can only be reliably typed in through the use of an assembler, such as Big Mac, and not everybody has one of those yet.

Applesoft's creators anticipated all of this when they included the Ampersand feature. Through that little device, additional commands could be added. They left it to the ingenuity of users to come up with ways of organizing the additional commands.

The five commercial packages we have reviewed give certain benefits to an Applesoft programmer, regardless of experience or lack of it. These benefits include: a PRINT USING command, Applesoft's biggest lack; a comprehensive package of commands which can be called with words and letters that imply on their face what they do; manuals with a how-to-use-it emphasis, rather than the why-I-wrote-it theme of so many magazine articles; tools to improve the operation of the user's programs themselves, as good tools already exist to assist in the writing of programs.

A beginning user gets the additional benefit of being able to use machine code right away, without having to learn how to write it, save it, load it, or hook it in. The packages we review do that for him.

Routine Machine and Amper-Magic provide a means of organizing an almost unlimited number of modules for the sake of a specific program, which can then be run on almost any Apple without the need for special booting procedures or master disks. The extensions they provide become part of the program.

Comparison of Program Features

TYPE	OPERATION	AMPER-MAGIC	AMPER-WARE	ROUTINE MACHINE	AMPER-MANAGER	AMPER-SOFT
I/O	Print Using	X	X	X	X	X
I/O	Format strings (left, right, center)	X	X		(6)	(6)
I/O	Dec to hex	X		X	X	
I/O	Break print lines on spaces	X		X		
I/O	Print upper case inverse and lower case normal on screen		X			
I/O	Set screen window	(3)	(2)	(3)		
I/O	HOME		(2)	(3)		
I/O	Move screen cursor to X,Y	(1)	(2)	(3)	X	
I/O	Input illegal characters	X	X	X	X	
I/O	Very sophisticated INPUT command		(4)	(3)		
I/O	Clear to line end	(3)	X	(3)		
I/O	Clear to page end	(3)	X	(3)		
I/O	Show ctrl chars in inverse			(3)	X	
I/O	Quick 33-col window				X	
I/O	Beep pitch, duration			X	X	
I/O	Hex to dec			X	X	
I/O	Print string as stored in memory				X	
I/O	Print char or string N times	(3)		(3)	X	
I/O	Sound effects			X		
I/O	Error messages without halting			X		
I/O	Print an entire array	(3)				X
I/O	Print a message on screen during DOS WRITE	(3)				
I/O	Set up peripherals with strings instead of POKES	(3)				
I/O	Send ESC codes to printer				X	
I/O	Text screen dump	(3)		(3)		
I/O	Formatted LIST				X	
I/O	Disconnect DOS for faster output				X	
FLOW	GOSUB [aexpr]	X	X	X	X	
FLOW	GOTO [aexpr]	X	X	X	X	
FLOW	RESTORE to a line number	X		X		
FLOW	RESTORE to a specific DATA element			X		
FLOW	Change program pointers to speed subroutine execution	X				
FLOW	ON CTRL CHR GOTO to branch if ctrl char input		X			
FLOW	Enter monitor					
FLOW	Wait N tenths of a second	(3)		X	X	
FLOW	Fix stack during error handling	(3)		X	X	

AMPERSAND UTILITIES

They come with a large assortment of modules from which to choose, and both publishers plan additional volumes of modules. These will tend to focus in an area, i.e. one disk of graphics tricks, another of fancy math, another for disk handling, and so forth. These would be the ones to buy in order to experiment with your own machine code extensions, or those found in the magazines.

To use either of them is easy, does not require knowledge of assembly language, but does involve extra steps for each program with which they will be used. Within limits, commands can be given any name the user likes. The problem here is that the name might not be consistent across all programs written by one programmer, and will almost certainly not be consistent from one programmer to another. This makes for extra work when going back over program listings some time after they were written.

The other packages, AmperSoft, Amper Manager, and Amperware, become as much a part of the Apple "system" as Applesoft and DOS. They extend the language by their presence. Their fixed syntax will be constant across all programs written by *all* programmers working with them. Unfortunately, they make it less than easy to include user-supplied command modules. Each one has special attributes. Amperware is a treat for data handlers. Amper Manager will have DOS enthusiasts turning cartwheels. AmperSoft looks like a winner in the math department.

Here is one idea for those who can't decide: get one of each. Routine Machine and Amper-Magic show the user how to locate the start of their "front-end" interface to appended modules. Assign that value to a variable, say, MODULE. Then substitute CALL MODULE for the Ampersand in accessing appended commands. Leave the Ampersand for the commands in Amper Manager, Amperware, or AmperSoft.

As we said, no "pick of the crop." But with the prices of hardware coming down so much, it is possible to pick up a more complete Apple than we could buy a year ago, yet spend no more than we did. Certainly, any one

Comparison of Program Features (Continued)

TYPE	OPERATION	AMPER-MAGIC	AMPER-WARE	ROUTINE MACHINE	AMPER-MANAGER	AMPER-SOFT
FLOW	CALL hex address				X	
FLOW	Reset ampersand to previous			X	X	
FLOW	ON RESET options such as boot, goto, run			X		
FLOW	Spacebar scroll pause control	(3)		(3)		
DISK	Binary file start and length into a variable without loading file	X		X		
DISK	Disk free space into a variable		X			
DISK	Save and load arrays as binary files (16 times faster)		X			X
DISK	Catalog				X	
	Catalog deleted files				X	
	Extended catalog				X	
	Undelete a file				X	
	Change drives				X	
	Binary file data after loading				X	
	Sophisticated DOS commands:				X	
	Take and use a DOS buffer by name				X	
	Use RWTS on said buffer, by name				X	
	Get and display the first sector of a file				X	
	INIT disk with a file name, but don't save				X	
	INIT disk without a DOS image				X	
	INIT selected tracks				X	
	Put a new DOS on disk				X	
	Put a new CATALOG on disk				X	
	Put a new VTOC on disk				X	
	Find and "hide" bad sectors				X	
	Modify the most recently used DOS buffer				X	
DISK	Faster BLOAD			X		
DISK	Access a file from catalog with two keystrokes					X
DISK	Fake a carriage return to DOS without writing it to disk	(3,5)				
MEM	Delete array	X	X	(3)		X
MEM	Disassemble memory	X		(3)		
MEM	Dump variable contents	X				
MEM	PEEK 16 bits	X		X		
MEM	POKE 16 bits	X		X	X	
MEM	POKE a series of bytes in hex or decimal	X		(3)	X	
MEM	POKE individual bits	(3)				
MEM	Memory move	X		X		
MEM	Hex dump memory	X		(3)	X	
MEM	Move a text screen image for instant recall	(3)				

Comparison of Program Features (Continued)

TYPE	OPERATION	AMPER-MAGIC	AMPER-WARE	ROUTINE MACHINE	AMPER-MANAGER	AMPER-SOFT
MEM	Swap variables	X		X	X	
MEM	Sort array		X	X		X
MEM	Fast garbage collection					X
MEM	Move DOS to language card to give 10,000 more bytes					X
STR	Substring search	X	X	X		
STR	Search string array	X	X	X		
STR	Reverse a string				X	
	Modify string bytes directly without elaborate string handling				X	
	Point a string descriptor at memory				X	
	Change string characters to flash, inverse, normal while in memory				X	
GRAF	Shape tables: relocatable, multiple on hand			X		
	Shape table viewer			X		
GRAF	Hi-Res ASCII character set			X		
GRAF	TURTLEGRAPHICS			X		
MATH	Powerful set of matrix functions, including:					
	Minima					X
	Maxima					X
	Inverse					X
	Transpose					X
	Identity					X
	Absolute Maxima					X
	Matrix multiplication					X

Footnotes

1) Works on screen only, 40–column

2) Works on 40 or 80–column screens

3) Available on an extra disk at extra cost. This was true for Amper–Magic at the time of writing. Disks with additional commands for The Routine Machine have also been released by the vendor.

4) Unique to Amperware, this INPUT command is very powerful.

5) DOS 3.3, 48K or 64K only.

6) Justifies strings left and right only.

of these packages will add value to a system. Adding an inexpensive assembler, say, Big Mac, completes the equipment list to handle almost any programming challenge. The rest comes with practice, practice, practice.

But that is another story.

EDITOR BYTES BACK

From Page 5

Who ever heard of doing card tricks on an Apple? We have, and so will you, when you read **Apple - The Magician** by returning author **Luis Hernandez.** Bah, humbug indeed!

Another **Call -A.P.P.L.E. technique** appears in this huge March issue; the subject: **Customizing GPLE Escape Tables;** the author: **Edward Nutter.**

Of course there's more, but we'd rather use our remaining space to clue you in on a couple of super stories you can look forward to over the next couple of months. **Michael J. O'Neill** is scheduled to show up soon with a **Telecommunications Tutorial, Burt Webb** has some interesting projections in what we can expect to find in the **Computer Home** any day now, **Gary Little** will present a method to spool your printer output, and **Chris Joachumson** will present **Arcade Graphics Techniques,** the first, we hope, of a series.

As a parting shot, we'll again ask you to drop a note to the editorial office letting us know your likes, dislikes and wants. As always, we're here for the benefit of our members, and our policies are guided by your thoughtful suggestions.

TWO UNIQUE HI-RES SYSTEMS!

THE ANIMATOR A system for producing **animated 'film' strips** with a minimum of work. You enter **only key frames** - The animator calculates the in-between frames to produce smooth motion. There is capacity for 50 independently moving parts in each frame. **The Animator** contains everything you need to create **animated games, choreograph dances,** make **logos,** or just **create cartoons.**

★ Demo includes ballet sequence (ballerina has 12 independently moving body parts).
★ Over **50 pages** of documentation, including **3 tutorials** & directions for adding strips to your own programs.
★ Powerful film editor includes: insert, delete, merge, translate, freeze frame, and many more.
★★ **BONUS!** A system for making shapes/shape tables - works independently of **The Animator.** It contains some unique features such as automatic construction of shapes at user defined angles.

SIGHT 'n SOUND Enhance any stereo or cassette recording with a **fully synchronized Hi-Res** light show. Use designs of your own creation or use those supplied. Features:

★ Kaleidscope light show ★ Guitar player who sings & plays to the beat ★ Geometric design light show.
★ 21 page clearly written manual tells:
- How to create light shows
- How **SIGHT 'n SOUND** works
- How to add light shows to your own programs

★ Watch your voice or 'EKG' on the oscilloscope
★ 9 predesigned light shows - ready to run
★ Powerful design editor for creating light shows
★ No hardware modifications.

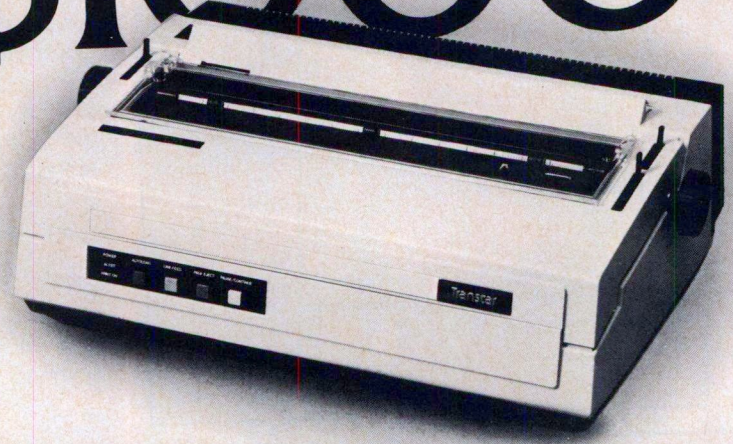
Both systems require a 48K Apple II+ or IIe or II with Applesoft and a disk drive.
Send check or money order to
BalbeSoftware Systems
#6 White Plains Dr.
St. Louis, MO 63017

THE ANIMATOR $49.95
SIGHT 'n SOUND $24.95

Please add $2 ($7 overseas) postage/handling
For information call (314) 532-5377
Dealer inquiries invited
Apple is a trademark of Apple Computer, Inc.

Circle 4 on Reader Service Card

Is 42 seconds worth $1000?



The new Transtar 130 daisy wheel printer generates a full-page letter in 78 seconds. The least expensive 40 cps printer does it in 36. Only 42 seconds difference...for twice the price.

At only $895, the Transtar 130 letter-quality printer makes speed its only compromise. Shannon-text rated at 16 cps, the Transtar 130 gives you better printing quality than any 40 cps printer. It allows you the full range of word-processing functions such as proportional spacing, superscript, subscript, underscoring and a true boldface. The 130 is "plug and go" compatible with the best-selling word processing packages. It's quiet: only 65dB. It's durable. It boasts a unique new autoload feature that automatically loads paper to one of four preselected positions with the touch of a button. And, as if that weren't enough, its end-user warranty runs a full six months—twice that of most of its competitors.

Affordable and loaded with all the features of printers costing twice as much, the Transtar 130 letter-quality printer retails for less than $900. But it's up to you: Is 42 seconds really worth $1000?

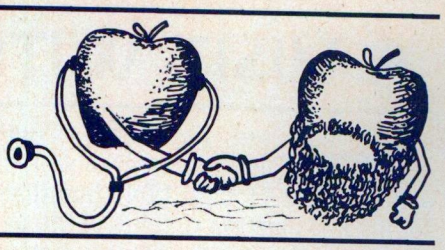
Transtar

Box C-96975, Bellevue, Washington 98009

Circle 5 on Reader Service Card

A VISIT WITH CORNELIS BONGERS

Ralph Swerdlow, M.D./Call — A.P.P.L.E. Staff Writer



BECAUSE of my interest in Apple Computers, I recently found myself having dinner in a suburban home just outside of the city of Rotterdam in The Netherlands. Our hosts were Cornelis Bongers, a fellow Staff Writer for *Call -A.P.P.L.E.*, and his wife, Margo.

My wife and I were in Europe on a combination business and professional trip. I was scheduled to attend a medical meeting in London, and we used this occasion to do some sight seeing on the continent.

Several weeks prior to leaving home, I mentioned our plans to Val Golding, and told him that we were planning on being in Amsterdam for several days. He reminded me that Dr. Bongers lived in Rotterdam, and suggested that I give him a call while in Europe. This we did, and I would like to describe our visit so that the readers of *Call -A.P.P.L.E.* may learn something about their staff writers.

I called Dr. Bongers from Amsterdam, and introduced myself. He recognized my name immediately, and said that I would not believe it, but at the moment I had called him, he was reading one of my articles in a past issue to review a specific point!! He then invited us to come to Rotterdam and have dinner with his family, giving me directions to his home. This involved a train ride across the country, and then a subway ride to a specific station, where he was to meet us. We were to call him when we arrived, so that he could pick us up and drive us to his house. He told me what he would be wearing so that I could identify him.

Our journey over, we emerged from the subway car, and as planned, there stood Dr. Bongers, a tall, slender man with a full flowing red beard dressed in the prescribed yellow windbreaker. He was accompanied by a large brown dog who seemed to be enjoying the outing.

The four of us, Dr. Bongers, my wife, the dog, and I, hopped into the car and away we went to the Bongers' home. We were in a suburb of Rotterdam, but except for the road signs being in Dutch, everything seemed to be about the same as here in suburban California. Off in the distance, our host pointed to the outlines of Erasmus University where he works. Five minutes later we were all sitting in the Bongers' living room, talking as old friends do. Our Apples bridged the gap between two countries and 6000 miles of distance!!

Dr. Bongers has a Ph.D in statistics, and teaches economics at the university. His wife, Margo, teaches history at a nearby high school. They are the parents of two young children, a boy aged three, and a girl less than a year old. Their boy was at the baby sitter for the evening, but their girl was sleeping soundly in her room. My wife was invited to look in at the sleeping infant, and her verdict was "just a beautiful baby."

The Bongers both speak English almost perfectly, he perhaps slightly better than she. They have been in the United States on several occasions, improving their English during these visits. Their initial knowledge, however, came from their schooling in The Netherlands, as English is a required subject. They did not speak Dutch in our presence, which made the evening very comfortable, for my wife and I did not know a single word in that language!

Mrs. Bongers is taking some time off from teaching, but will resume as soon as her girl is slightly older. She is very proud of her husband, and showed us his Ph.D thesis, which she took from a huge bookcase that covered an entire wall of their living room. It is a large textbook, full of advanced mathematics, and equations. He told us that he did not use an Apple in its preparation, but wished he had one available when he wrote it. "Would have made things much easier," he adds.

Bookcase contents indicate something about the users, so I looked at the books more carefully. There were books on travel, history, literature, mathematics, and technical books all in several languages. They are both obviously well read, and interested in a great number of subjects, and they both admitted an interest in Central American history.

Soon the conversation turned to computers, as might be expected. He spends a great deal of time "playing with his computer," according to Mrs. Bongers. My wife joins in on this theme with a great deal of enthusiasm. It seems that I, too, spend "too much time" with my Apple, and that both wives would like their respective husbands "to get out doors more," and "spend less time in their rooms with their toys." Both wives are proud of their husbands' being writers for *Call -A.P.P.L.E.* but don't understand computers very well, are somewhat bored by computer talk, and the compulsion demonstrated by their spouses. (Does this sound familiar to you??)

Dr. Bongers and I then started comparing notes about how we write and how we get the ideas for stories. It is amazing how similar our approaches are. It seems that we both do so very much in the same way. An idea occurs to us, and then we spend time thinking about the problem. Not time at the computer; just thinking about something that either puzzled us or to meet a need that we have at the moment. It is only later that we sit down at the computer to work out the problem.

First comes extensive testing of the program, written in assembly language or BASIC, until we are reasonably certain that all of the major bugs have been found and corrected. We both admitted that there is always a small, but realistic fear that a bug will surface after we write the story. It is only after this process that the story begins to be typed. He said that he writes his stories on a normal typewriter. I use my computer and a word processor program. He added that he is able to write very rapidly, with a minimum of corrections. The same is true in my case. He proof reads his stories and makes corrections with his typewriter. I use a spelling program, and printout several "hardcopies," and make corrections with my word processor.

A VISIT WITH BONGERS

We discussed the source of our ideas, and again it is remarkable the similarities we found. I told him that for me it is hard *not* to think of things to write about, and that every computer journal raises questions that I need answered and work on them until I understand. He too goes through an identical thought process, but is more concerned with assembly language programs than am I.

In fact, he added that he started to do machine language programming about three days after he obtained his computer. It seems that the machine would not do something he asked it to do, so he had to "fix it" by writing a section of machine code. He graduated to an assembler shortly afterward and tried several commercial programs then available. He now uses *Big Mac* exclusively. "It is the best and fastest assembler on the market now, and it is easy to use and to understand."

He is a person who sees the world of computers only in black or white. He said frequently, as we talked, "That is good or that is terrible." No real in between, but a great deal of self certainty in his opinion as to the present state of computing, and of the software available.

He is dead set against copying commercial programs, but thinks that most software is over priced for what the consumer has to pay for the package. (I agree with him totally.) He said that he has never given a copy of any program that he purchased to any one. What he writes himself he shares if it seems appropriate. He added that he does not accept copies of commercial programs from friends either.

We then talked about computers in Europe, and he said that it is a very active avocation, both for recreation and for serious work. There are many users groups and many of the university staff have their own Apples which they use in their professional work.

He predicted that the next few years would see prices fall, and better software appear. He is interested in commercial applications for some of the things he is writing, and feels that they could serve a useful purpose.

I mentioned that I am interested in hard disks now, and have been looking at what is available. He is also concerned about having a larger memory system, but has not yet decided on which system to adapt. He feels that hard disks will become less expensive and will wait for several months to see which way things go.

This was the way the conversation went while we were eating. The wives listened to us talk. We were delighted with what we were saying; the women were bearing up with just a slight twinge of pained expression on their faces. So prudence dictated a change of pace, at least for a while.

Mrs. Bongers is a magnificent cook, and had obviously expended some effort to prepare a special dinner for us. Dr. Bongers and I then changed the tone of the conversation away from Apples, and we talked about every day things, including life in the U.S. and in the Netherlands. We compared university teaching, medical practice, cost of living and had a marvelous time doing so. We had promised ourselves that after dinner, we would go to Dr. Bongers' computer room and "play" (as the wives said) with his new "toy," a Basis 108. (See the August 1982 issue of *Call –A.P.P.L.E.* for the article he wrote about this machine.)

After dessert, he and I went up to his study. It is a large room, kept in obvious controlled disarray. He knew where everything was, and could reach for any disk with great facility. He keeps his disks in a small file box like the one my wife uses to store her recipes. He has a large desk (so do I) and his computer is off to one side of the desk. He has two disk drives, and a small monitor set on top of his computer. He handled the computer easily and with great confidence. He showed me the Basis 108, how it can do all the things described in his article.

I was impressed by this machine possessing, in contrast to the Apple][, built–in numeric keypad with cursor control keys as well as digits. It took me several minutes to adjust to reaching for the keypad instead of the arrow keys.

He had several back copies of *Call –A.P.P.L.E.* laying on his desk, and they were obviously well read. He also had a book case with a collection of computer text books close at hand.

It was becoming late, and Margo informed us that we would miss the last train to Amsterdam unless we left for the train station very shortly. Our visit was over, much too soon for me. He drove us into Rotterdam, and we caught the last train out.

I hope that when the Bongers come to the U.S., they will visit us. They are really such nice and warm people and there are so many additional questions that I would like to ask Cornelis. It turns out that *Call –A.P.P.L.E.* staff writers are normal humans after all, as I trust this report demonstrates.

Emulates these terminals exactly.

IBM 3101
DEC VT100, VT52
Data General D200
ADDS Regent 20, 25, 40
Hazeltine 1400, 1410, 1500
Lear Siegler ADM-3A, ADM-5
TeleVideo 910
Teletype Model 33 KSR

Apple is a trademark of Apple Computer, Inc.

SOFTERM®

High-speed CRT Look-alike Software for Your Apple®

New File Transfer Language

BREAK
CATALOG
CHAIN
CONFIGURE
CONNECT
CONVERSE
DIAL
END
HANGUP
LOG
MONITOR
NOLOG
ONERR
PAUSE
PROMPT
RECEIVE
REMARK
RETRIES
SEND
SPECIAL
SPEED
TIMEOUT
XMIT:WAIT

Supports these interface boards.

Apple Communications Card
Apple Parallel Printer
Apple Serial Interface
Apple Super Serial Card
Bit 3 Dual-Comm Plus™
CCS 7710,7720, 7728
Hayes Micromodem II™
Hayes Smartmodem™ 300, & 1200
Intra Computer PS10
Mountain Computer CPS Card™
Novation Apple-Cat II™ 300 & 1200
Orange Micro Grappler™
Prometheus VERSAcard™
SSM ASIO, APIO, AIO, AIO II™

Supports your 80-column hardware.

ALS Smarterm™
Bit 3 Full-View 80™
Computer Stop Omnivision™
M&R Sup'R'Terminal™
STB Systems STB-80™
Videx Videoterm™
Vista Computer Vision 80™
Wesper Micro Wizard 80™

Your host computer won't know the difference!

Softerm provides an *exact* terminal emulation for a wide range of CRT terminals which interface to a variety of host computer systems. Special function keys, sophisticated editing features, even local printer capabilities of the terminals emulated by Softerm are fully supported. Softerm operates with even the most discriminating host computer applications including video editors. And at speeds up to 9600 baud using either a direct connection or any standard modem.

Unmatched file transfer capability

Softerm offers file transfer methods flexible enough to match any host computer requirement. These include *character protocol* with user-definable terminator and acknowledge strings, block size, and character echo wait, and the intelligent *Softrans*™ protocol which provides reliable error-free transmission and reception of data. The character protocol provides maximum flexibility for text file transfers. *Any* type file may be transferred using the Softrans protocol which provides automatic binary encoding and decoding, block checking with error recovery, and data compression to enhance line utilization. A FORTRAN 77 source program is supplied with Softerm which is easily adaptable to any host computer to allow communications with Softerm using the Softrans protocol.

Softerm file transfer utilizes an easy to use *command language* which allows simple definition of even complex multiple-file transfers with handshaking. Twenty-three high-level commands include *DIAL, CATALOG, SEND, RECEIVE, ONERR, HANGUP, MONITOR* and others which may be executed in immediate command mode interactively or from a file transfer macro command file which has been previously entered and saved on disk.

Built-in utilities

Softerm disk utilities allow DOS commands such as CATALOG, INIT, RENAME, and DELETE to be executed allowing convenient file maintenance. Local file transfers allow files to be displayed, printed, or even copied to another file without exiting the Softerm program. Numerous editing options such as tab expansion and space compression are provided to allow easy reformatting of data to accommodate the variations in data formats used by host computers. Softerm supports automatic dialing in both terminal and file transfer modes. Dial utilities allow a *phone book* of frequently used numbers to be defined which are accessed by a user-assigned name and specify the serial interface parameters to be used.

Online Update Service

The Softronics Online Update Service is provided as an additional support service at no additional cost to Softerm users. Its purpose is to allow fast turnaround of Softerm program fixes for user-reported problems using the *automatic patch facility* included in Softerm as well as a convenient distribution method for additional terminal emulations and I/O drivers which become available. *User correspondence* can be electronically mailed to Softronics, and *user-contributed* keyboard macros, file transfer macros, and host adaptations of the Softrans FORTRAN 77 program are available on-line.

Most advanced communications software available

Just check Softerm's 300 page user manual. You simply can't buy a more sophisticated package or one that's easier to use. Available now for only $150 from your local dealer or Softronics, Inc.

SOFTRONICS

6626 Prince Edward. Memphis. TN 38119. 901-755-5006

Circle 35 on Reader Service Card

CONGRATULATIONS.

YOUR VISICALC™ JUST DOUBLED IN VALUE.

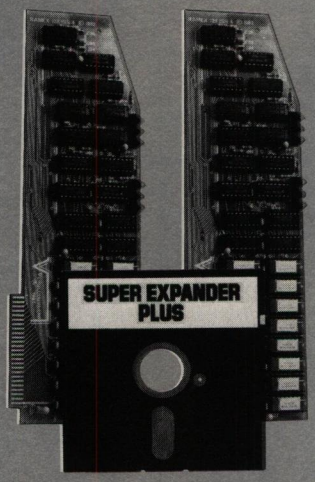
New Super Expander Plus™ pre-boot for Ramex-128™ board upgrades any Apple II™ 16-sector VisiCalc™ to look like the Advanced Version.

SUPER EXPANDER PLUS keeps all your existing VisiCalc facilities, and adds variable column width, global formatting of numbers, negative numbers in brackets, password protection, new format commands, tabbed fields, the works. It even supports an 80-column card if one is present.

If 136K is enough, you can get all these features with just one Ramex-128 board. But SUPER EXPANDER PLUS supports **two** Ramex-128K cards to give you an incredible 255K VisiCalc File, and dumps the whole model back and forth to floppies **in less than 40 seconds**.

There's simply **no** other memory-expansion/software combination that even comes close. All the others give you less memory, take longer to load and save (as much as fifteen minutes longer), and cost more for what you get. You've got to see it to believe it. Quick! Call your dealer.

Ramex-128 card, just $499.
SUPER EXPANDER PLUS, just $125.



SUPER EXPANDER PLUS™

OMEGA MICROWARE, INC. T.M.
222 SO. RIVERSIDE PLAZA
CHICAGO, IL 60606
312-648-4844

MasterCard and Visa holders order **toll-free 1-800-835-2246**

Following our long-established Omega MicroWare policy, present Super Expander-40 and -80 owners may upgrade for the price difference only. Phone our office for details.

Apple is a registered trademark of Apple Computer, Inc. VisiCalc is a registered trademark of VisiCorp, Inc. Super Expander Plus, Ramex-128, and Omega MicroWare are trademarks of Omega MicroWare, Inc.
© 1983 Omega MicroWare, Inc.

Circle 33 on Reader Service Card

ESCAPE FROM THE MOTHERBOARD

Donald W. Miller, M.D.

INSPIRED by *Call A.P.P.L.E.'s* recent articles on potential uses of a RAM card, I dusted off my *Apple][Reference Manual* as well as the language card manual. For the past year and a half, my RAM card has seen little action, except to run Integer Basic programs (rarely) or occasionally the RAM chips werre rustled by a program (Visicale, Big Mac, etc.) that made use of the extra memory.

First, let me say that I've never been really interested in the plethora of utility programs that have been published in various magazines. I mean such things as, ampersand utilities, autonumbering, and (please, if I see another one, I'll croak!) *Print–Using* implementations. Most of these have to be complexly interfaced and intertwined with the existing software, and, at least to me, don't seem practical.

What excited me, was the tidbit dropped by Craig Peterson in the December, '82 *Call A.P.P.L.E.* There it was mentioned that the actual Applesoft and monitor code could be modified (I assume by, disassembling, modifying, and then reassembling). This new "language" could then be placed on the RAM card, much like "foreign" languages are placed there now. This is what woke me up one Saturday morning and then consumed the rest of my day.

To make any modifications to the monitor, first a labile, fluid RAM version of the monitor (and Applesoft language) is needed. My initial experiment to create this version consisted of BSAVEing the entire memory area $D000–$FFFF to the disk, then enabling the RAM card, and finally BLOADing (writing) the file to the card. This satisfied my requirements since I then had the monitor routines in modifiable RAM. However, the thought of loading this long (52 sectors) file into memory each time, just to change a dozen or so bytes, was more than I could stand.

About one o'clock the same afternoon, I discovered that the ROM's from $D000–$FFFF could be directly written to the RAM card, even using the monitor's MOVE routine at the same time. It seemed peculiar to me that the Apple would read a byte at one location (in ROM), and then write to the SAME ADDRESS (RAM) at a different physical location. But, what do I know?

To do this in the immediate mode, the RAM card must first be write enabled, but the ROM's must still be active. This is done by consecutively accessing location $C081(49821). Next, after entering the monitor (CALL –151), type D000<D000. FFFFM. The entire move takes place in less than a second. Now, access $C083(49283) and the Apple is running from the RAM card. At this point, any memory location can be changed, either from the monitor, or by POKEing from BASIC.

Now, what to modify?

An article in the Nov–Dec, 1980 *Call A.P.P.L.E* came to mind. This article covered the case of the NFC (Non–Flashing Cursor). Basically, after dissecting the code, only two bytes ($FD13,$FD14) would have to be changed to produce the NFC. The trick at that time was to get around the inability to directly change the monitor. The best solution then was to modify DOS. Now with the monitor in RAM, this can now be easily accomplished and *voila,* a NFC!

How about changing the bell tone?

Why not? Just change the byte at $FBE5 (15($0F) is nice!).

If you've seen an Apple *III* in action, you may have noticed that when the [ESC] key is pressed, the cursor changes to an inverse "+" and remains that way, as long as the subsequent keys pressed are legal escape keys. Now the Apple][can have this same feature, and here's how.

As you may recall from NFC, the flashing cursor is actually generated by the RDKEY routine. The routine is as follows:

```
RDKEY   LDY CH
        LDA (BASL),Y
        PHA
        AND #$3F
        ORA #$40
        STA (BASL),Y
        PLA
        JMP (KSWL)
```

If we removed the AND and ORA expressions, and replaced them with

```
LDA #$6B ASCII Screen
         code for
         flashing "+"
```

the cursor would become a flashing "+" all the time. However, this new cursor is only needed when the [ESC] key is pressed. Digging further into the monitor code, and looking particularly at the escape routines, RDKEY is called at locations $FD2F and $FBA2. To solve the problem, these two locations should probably call a second RDKEY routine, RDKEY2.

```
RDKEY2  LDY CH
        LDA (BASL),Y
        PHA
        LDA #$6B
        STA (BASL),Y
        PLA
        JMP (KSWL)
```

Fortunately, there also happens to be a "hole" of 14 consecutive NOP's in the autostart monitor code, right below the actual escape routines! RDKEY2 is the perfect choice to fill this location ($FBB3). The previous escape calls to RDKEY can now be changed to CALL RDKEY2 and the job is done.

The following program when BRUN, will make all the necessary moves and changes. Total time to BLOAD and BRUN this program is less than three seconds. The patch section of the code is meant to be general so that longer routines can be patched into the RAM monitor. More room can be created by discarding rarely used routines, such as VERIFY. User designed routines, or even previously awkward utilities, can then be used easily.

ESCAPE FROM THE MOTHERBOARD

```
                   ********************************
                   *                              *
                   *  Escape From the Motherboard *
                   * by  Donald    Miller Jr., MD *
                   *                              *
                   * Call -A.P.P.L.E. : Mar. 1983 *
                   *                              *
                   ********************************
                   *
                            ORG  $300
                   *
                   DEST     EQU  $42        ;A4L *
                   START    EQU  $3C        ;A1L * Used in move routine
                   END      EQU  $3E        ;A2L *
                   MOVE     EQU  $FE2C      ;Monitor move routine
                   ENABLE   EQU  $C081
                   READ     EQU  $C083
                   LOC      EQU  $D000      ;Applesoft & monitor start
                   NEWCODE  EQU  $FBB3      ;Patch location
                   CODELOC  EQU  $6
                   NOP      EQU  $EA
                   *
                   *
                   *
0300: A9 00                 LDA  #00        ;Initialize
0302: 85 3C                 STA  START
0304: 85 42                 STA  DEST
0306: A8                    TAY             ;Must zero Y before move
0307: A9 FF                 LDA  #$FF       ;End of memory ($FFFF)
0309: 85 3E                 STA  END
030B: 85 3F                 STA  END+1
030D: A9 D0                 LDA  #/LOC
030F: 85 43                 STA  DEST+1
0311: 85 3D                 STA  START+1
                   *
                   *
0313: AD 81 C0              LDA  ENABLE     ;Consecutive accesses for
0316: AD 81 C0              LDA  ENABLE     ;RAMcard write
                   *
                   *
0319: 20 2C FE              JSR  MOVE       ;Monitor move
                   *
                   *
031C: AD 83 C0              LDA  READ       ;Select RAMcard
                   *
                   *
031F: A9 B3                 LDA  #NEWCODE   ;Start patching
0321: 85 06                 STA  CODELOC
0323: A9 FB                 LDA  #/NEWCODE
0325: 85 07                 STA  CODELOC+1
0327: A0 00                 LDY  #00
0329: B9 54 03     CONT     LDA  RDKEY2,Y   ;Write RDKEY2
032C: 91 06                 STA  (CODELOC),Y
032E: C8                    INY
032F: C0 0D                 CPY  #13        ;RDKEY2 length
0331: F0 03                 BEQ  CONT1
0333: 4C 29 03              JMP  CONT
0336: A5 06        CONT1    LDA  CODELOC    ;Continue patching...
0338: 8D A3 FB              STA  $FBA3
033B: 8D 30 FD              STA  $FD30
033E: A5 07                 LDA  CODELOC+1
0340: 8D A4 FB              STA  $FBA4
0343: 8D 31 FD              STA  $FD31
0346: A9 EA                 LDA  #NOP       ;Non-flashing cursor
0348: 8D 13 FD              STA  $FD13
034B: 8D 14 FD              STA  $FD14
034E: A9 0F                 LDA  #15        ;Change bell tone
0350: 8D E5 FB              STA  $FBE5
0353: 60                    RTS
0354: A4 24 B1
0357: 28 48 A9
035A: 6B 91 28
035D: 68 6C 38
0360: 00           RDKEY2   HEX  A4,24,B1,28,48,A9,6B,91,28,68,6C,38,00
--End assembly--   97 bytes     Errors: 0
```

APPLE — THE MAGICIAN

Luis M. Hernandez

THIS program is a representation of an old card trick which does not fail to amuse both young and old. I will describe the trick instead of the program, so that aside from learning a card trick, you readers out there can also get some practice in exercising the logic circuits of your mind.

In order to visualize the trick, pick out any 21 cards from a 52-card deck. Deal out the first three cards *horizontally,* setting up three columns, then deal out the next three cards (horizontally again) to form the next row of cards (the cards must rest over but not completely covering the previously dealt cards). Then deal out the rest of the cards in the same manner.

After doing so, you will have three columns with seven cards each. Now ask a friend to pick out a card and, without telling you what it is, indicate which column contains the card. Knowing the column, you in turn will take the seven cards of each column and form them into three stacks, that is, the seven cards in column one will form a stack, column three another stack, and so on). Now when you put the three stacks together, make sure that the stack from the column which your friend indicated goes in the *middle.* For example, if your friend indicated column three, the whole seven cards from this column or stack should go between the seven cards each of columns one and two (which of column one or two goes to the top or bottom is irrelevant). This is the whole trick. The chosen column should go in as the middle stack.

Next you deal out the 21 cards for the second time. Again ask your friend to indicate which column contains his/her card. After having done so, pick up the 21 cards as before but again putting the stack of the indicated column in the middle. Deal the 21 cards again, for the third and last time, in the same manner as before and ask your friend to indicate which column contains his/her card. You, on the other hand, should fake a lot of face-distorting thinking at this point, holler out a lot of nonsense mumbo-jumbo, and then slowly pick out the fourth card from the top of the column your friend has indicated. It works every time, believe me.

When you understand how the trick works, the program will surely be a cinch.

Have fun, folks!!!

```
]LIST

10  REM

    APPLE - THE MAGICIAN
    BY LUIS M. HERNANDEZ

CALL -A.P.P.L.E. : MAR. 1983

15  IF  PEEK (978) = 190 THEN 140

20  DIM A(21),B(21),DECK$(52),A$(
    21)
25  TEXT
30  FOR X = 1 TO 52
40  READ DECK$(X)
50  NEXT X
60  FOR X = 1 TO 21
70 T =  INT ( RND (1) * 53)
80  IF T = 0 THEN 70
90  FOR Y = 1 TO X - 1
100  IF A(Y) = T THEN 70
110  NEXT Y
120 A(X) = T
130  NEXT X
140  HOME
150 A$(0) = "APPLE - THE MAGICIAN
    "
160  PRINT  SPC( (40 -  LEN (A$(0
    ))) / 2);A$(0)
170  PRINT
180  FOR I = 1 TO 3
190  TEXT
200  FOR X = 1 TO 21
210 A$(X) = DECK$(A(X))
220  NEXT X
230  POKE 34,1
240  HOME
250  TEXT
260  VTAB 4
270  PRINT "COLUMN #1"; TAB( 14);
     "COLUMN #2"; TAB( 28);"COLUM
     N #3": PRINT : PRINT
280  SPEED= 200
290  FOR X = 1 TO 21 STEP 3
300  PRINT A$(X); TAB( 14);A$(X +
     1); TAB( 28);A$(X + 2): PRINT

310  NEXT X
320  VTAB 23
330  SPEED= 255
340  PRINT "INPUT COLUMN # CONTAI
     NING YOUR CARD:";: GET A$(0)

350 X =  VAL (A$(0))
360  IF X < 1 OR X > 3 THEN 320
370 F = 0
380  IF I = 3 THEN 530
390  FOR Y = 1 TO 3
400 L = 1
410  IF F = 1 THEN L = 15
420  IF Y = X THEN L = 8
430  FOR J = Y TO Y + 18 STEP 3
440 B(L) = A(J)
450 L = L + 1
460  NEXT J
470  IF Y <  > X THEN F = 1
480  NEXT Y
490  FOR Y = 1 TO 21
500 A(Y) = B(Y)
510  NEXT Y
520  NEXT I
530  POKE 34,1
540  HOME
```

The Pizza Program

Have you ever come home and wondered, "What are we having for dinner?" Now there's a great time saver for anyone who cooks or shops at the grocery store. Announcing the first dinner menu planning system.

ENDS HO-HUM DINNERS

Are you tired of the same old thing for dinner? Would you like more variety in your evening meal? Is there something you'd rather have but don't get very often? The Pizza Program is designed just for you. It's a delightful new software package designed to end the dinner-blahs with computer generated menus. Here is how it works.

You review what you like from the pre-selected food groups in the system. Delete any foods you don't enjoy. Add anything new at any time. Then decide how often you like to eat certain items. For example. Don't like liver? Then eliminate it with a few simple keystrokes. Or, you can plan for it as seldom as once every 99 weeks or as often as daily.

Want to go out to your favorite restaurant? Enter the restaurant's name as a "Main Course." Now your computer will automatically remind you to go out to eat—and as often as you select. It will delete all other items from that meal except the name of the restaurant.

AUTOMATIC SHOPPING LIST

You get a new menu each week or for just a few days if you want. And, it generates a detailed shopping list, automatically. It can arrange each item on the list in sequence according to the aisles at your favorite store. Studies show a shopping list will discourage impulse buying and save you money.

Also, it generates a per serving calorie counter. This is easy to delete anytime you are not in a diet mood or want to celebrate for any reason. You never count calories unless you want to.

RANDOMLY DELICIOUS

Say goodby to boring meals. Your computer will remember variety is the spice of life. This system makes eating at home a pleasure again. Each menu is randomly generated from major food groups according to the specific criteria you select. The system is easy to learn and easy to operate. Yet it is a sophisticated piece of software which will prevent menu mix-ups.

The Pizza Program is not a recipe file. Instead, it is a complete menu planning system you customize to fit your tastes and budget. It's a practical and easy way to organize your meal planning. You'll appreciate it day after day, week after week. People across the country are finding it a super idea—a great companion to any kitchen.

TRY IT FOR 30 DAYS WITHOUT RISK

You can try it with no obligation. Our home trial lets you actually use The Pizza Program for a full 30 days before you decide to keep it. Watch the fun and convenience it creates. Enjoy better meals and see how much time it saves. If you are not satisfied for any reason, return it within 1 month for a prompt and courteous refund. Your investment is just $34.50 plus $2.00 for shipping and handling. (California residents add 6½% sales tax.) A full 40 pages of instructions are included. And, it will be shipped promptly by first class mail.

OUT OF THE RUT

One woman's reaction to this program is typical. She wrote, "Before using your system I found myself getting into a rut of serving the same things over and over. The Pizza Program has changed all of this for me. We now have a wider variety of dinners and best of all I don't have to decide what they will be. If this was all it did, I'd be thrilled. But it isn't. The shopping list I receive along with my menus has been such a time saver. I quickly run through it and delete anything I feel I don't need and add something I might. I would have a hard time going back to doing my menus by hand."

This system requires an Apple II Plus* with 48K and 1 disk drive. (A printer is optional). We urge you to take advantage of our no-risk, 30 day home trial offer. To order call toll free and use your VISA or MasterCard. Or, send a check to the address below. There's no obligation. Order today.

ORDER TOLL FREE
(24/hrs/day)
800-453-4000
(In Utah call **1-800-662-8666**)

Gourmet Software
671 Eden Avenue
San Jose, CA 95117

*Apple is a registered trademark of Apple Computer, Inc.

THE MAGICIAN

```
550  TEXT
560  VTAB 11
570  PRINT "YOUR CARD IS ";A$(X +
     9); CHR$ (7); CHR$ (7); CHR$
     (7)
580  PRINT : PRINT "ANOTHER GAME?
     ";: GET A$(0)
590  IF  LEFT$ (A$(0),1) = "Y" THEN
     60
600  END
610  DATA  2-CLUBS,3-CLUBS,4-CLUB
     S,5-CLUBS,6-CLUBS,7-CLUBS,8-
     CLUBS,9-CLUBS,10-CLUBS,J-CLU
     BS,Q-CLUBS,K-CLUBS,A-CLUBS
620  DATA  2-SPADES,3-SPADES,4-SP
     ADES,5-SPADES,6-SPADES,7-SPA
     DES,8-SPADES,9-SPADES,10-SPA
     DES,J-SPADES,Q-SPADES,K-SPAD
     ES,A-SPADES
630  DATA 2-HEARTS,3-HEARTS,4-HEA
     RTS,5-HEARTS,6-HEARTS,7-HEAR
     TS,8-HEARTS,9-HEARTS,10-HEAR
     TS,J-HEARTS,Q-HEARTS,K-HEART
     S,A-HEARTS
640  DATA  2-DIAMONDS,3-DIAMONDS,
     4-DIAMONDS,5-DIAMONDS,6-DIAM
     ONDS,7-DIAMONDS,8-DIAMONDS,9
     -DIAMONDS,10-DIAMONDS,J-DIAM
     ONDS,Q-DIAMONDS,K-DIAMONDS,A
     -DIAMONDS
```


"My Legend memory expansion cards enhance my Apple and are a superb product for the Apple Computer.*

I think the cards are mapped appropriately for extra storage.

The software supplied makes the cards useful even for those who can't modify DOS on their own.

I look forward to upcoming products from Legend."

Steve Wozniak
Co-Founder,
Apple Computer, Inc.

If Steve Wozniak, Co-Founder of Apple Computer, Inc., thinks this highly of Legend products, we're doing something right.

Go to your nearest computer dealer today and find out for yourself about Legend products.

The Original Hi-Density RAM Products for APPLE II

Legend Industries Ltd.

U.S.A.
Legend Industries, Ltd.
2220 Scott Lk. Rd.
Pontiac, MI 48054
(313) 674-0953

WEST GERMANY
Porter and Co.
Berger Strasse 140
6000 Frankfurt/Main 60
West Germany
49 611-439818

EUROPE
B.I.P.
22 Joseph Disjon
75018 Paris, France
(1) 255-4463.

*Apple II is a product of Apple Computer, Inc.

Circle 38 on Reader Service Card

☑ D.R.S.™ Can Do It!

Nothing else even comes close to D.R.S.— our Disk Recovery System—not Bag of Tricks,™ not Dark,™ not Disk Fixer™—nothing!

☑ D.R.S. Can Do It! Put your damaged disk in one drive, a blank disk in the other. D.R.S. will do what all those other guys will do...and *far more!*

☑ D.R.S. Can Do It! No other program actually reads information from damaged sectors, but D.R.S. does.

☑ D.R.S. Can Do It! No other program can read disks that were recorded off-center, but D.R.S. does.

☑ D.R.S. Can Do It! No other program can read disks with errors caused by hub damage, but D.R.S. does.

☑ D.R.S. Can Do It! No other program types, grades and fixes all possible files, but D.R.S. does.

☑ D.R.S. Can Do It! No other program makes recoverable files accessible for conventional editing, but D.R.S. does.

☑ D.R.S. Can Do It! And all of this is totally automatic. D.R.S., a step ahead of the state-of-the-art in Apple disk recovery systems.

$100

SOF/SYS, INC.

4306 Upton Ave. So. • Mpls., MN 55410

Available at your local dealer or call us at

(612) 929-7104.

We accept VISA/ Mastercard

Circle 39 on Reader Service Card

THOUSANDS OF INNOVATIVE APPLE-COMPATIBLE PRODUCTS ARE GATHERING AT

Anaheim Boston San Francisco

Produced by Northeast Expositions, nationwide producers of the National Computer Shows, PC '83 and CP/M'83,
826 Boylston Street, Chestnut Hill, Massachusetts 02167.

DON'T MISS APPLEFEST FOR 1983 THE WORLD'S LARGEST EXPOSITION EXCLUSIVELY FOR APPLE OWNERS

Each show features hundreds of exhibits of the newest, state-of-the-art products for the Apple. You can see and try out software for every conceivable application—from arcade games to investment programs, music to machine language, teaching systems to accounting packages, word processors to graphics processors. You can sample hundreds of different peripherals, including printers, hard disks, modems, memory cards, video displays and synthesizers, plus accessories, publications and invaluable support services.

Applefest is **the** place to view the most technologically advanced products for the Apple.

At Applefest you can try out and compare hundreds of products in an exciting, information-filled environment. You can learn more in two days than you could in months of visiting computer stores and reading trade journals.

And, best of all, everything on display at Applefest is for sale at special show prices, so you can save hundreds—even thousands—of dollars by making your purchases at the show.

This year a whole new conference program is being introduced to Applefests nationwide. The program will show you how to squeeze absolutely the most power, versatility and usefulness out of your Apple.

Seminars and workshops will teach you the ins and outs of buying software intelligently, using spreadsheet and database programs, putting Apples to work in classrooms and using the Apple as a management tool. You'll learn about new programming languages, important applications for telecommunications, exciting ways to use graphics and more.

No matter what you do (or want to do) with your Apple, the Applefest seminars and workshops will help you do it better.

Software Spotlights will provide an in-depth, understandable look at hundreds of different software packages. Each Spotlight will cover the features, capabilities and limitations of a group of packages, to help you find the software that's best suited to your applications. Experts will be on hand to answer all your questions.

So plan on attending Applefest for 1983—the biggest and best Apple-user show ever. It'll be a mind-expanding experience for both you **and** your Apple.

Daily Registration Fees
Exhibits-only badges are $8 per day, and the Conference Program is $15 per day.

Special Pre-Registration Discount
If you plan to attend Applefest save now with advance registration. Three-day Exhibits and Conference badges are $48, you save $21. Three-day Exhibits only badges at $18, you save $6.

Additional Information
To receive more information about attending or exhibiting at Applefest, including the Conference, Seminar, Workshop and Panel Discussions Program, or information on local hotels call 617-739-2000 or 800-841-7000 (Boston).

apple Fest 1983

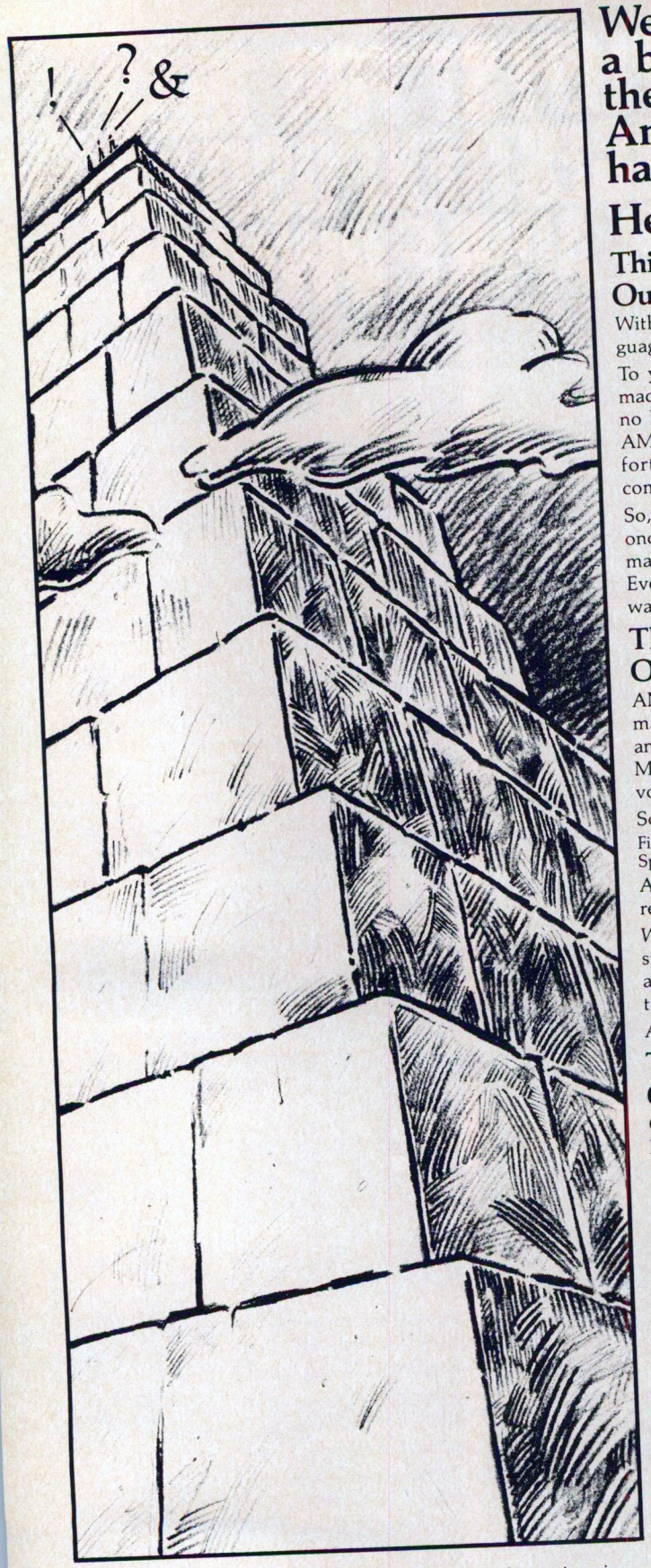
Applefest/Anaheim:
Friday-Sunday, April 15-17, 1983
Anaheim Convention Center
10:30AM-5:30PM daily

Applefest/Boston:
Friday-Sunday, May 13-15, 1983
Bayside Exposition Center
10:30AM-5:30PM daily

Applefest/San Francisco:
Friday-Sunday, October 28-30, 1983
Moscone Center
10:30AM-5:30PM daily

Apple and the Apple logo are registered trademarks and service marks of Apple Computer, Inc.

Any Questions? Call 800-841-7000 (Boston)



We've moved ahead a byte in time since the Tower of Babel. And at least three things have improved.

Here they are:

Thing Number One: Our Communication's Improved.

With AMPER-MAGIC™ there's no need to know machine language to benefit from its power.

To your Applesoft™ programs you can attach slick, finished machine language routines (referenced by name, not by address, no less) in seconds. No separate BLOADing. And then, with AMPER-MAGIC commands, you can pass variables back and forth directly, just as you would with built-in Applesoft commands.

So, name your routine, perform the append procedure (only once please), and AMPER-MAGIC takes over — generating machine language speed where it counts: in *your* programs. Even your commercial programs. And, if for some reason you want to remove the routine, AMPER-MAGIC can do this too.

Thing Number Two: Our Communication's Improved.

AMPER-MAGIC also gives you access to hundreds of relocatable machine language routines, including those from magazines and other libraries, and - most notably - those from AMPER-MAGIC itself, which has over fifty routines of its own in two volumes.

Some of AMPER-MAGIC's routines are:

Find substring	Move memory	Multiple poke
Speed up Applesoft	Input anything	Swap variables

And, of course, there are others too, for business, educational, recreational and word-process uses.

With AMPER-MAGIC you attach and access. It's really that simple. You can attach an unlimited number of commands to any program and you can move any set of commands as a unit to any other program.

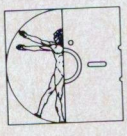
A marvelous simplicity, wouldn't you say?

Thing Number Three: Our Communication's Improved.

Compatible with AMPER-MAGIC you can get the AMPER-MAGIC COMMAND LIBRARY. Already available is Volume 2, which contains twenty-seven machine language routines devoted to information displays and output capabilities.

With Volume 2 you get: a WAIT command, which specifies time in ordinary tenths of seconds either as a variable or as an expression which can be controlled by the program; the most powerful PRINT USING command yet devised; and, of course, twenty-five other commands. Volume 3 will be devoted to Memory Management including a way to store 100,000 numbers in a 48K Apple.

Imagine!
Only $75.00 for the original AMPER-MAGIC
Only $35.00 for AMPER-MAGIC COMMAND LIBRARY, Volume 2
See your dealer or contact us.



Anthro-Digital

"The Company That Interfaces People and Computers"
103 Bartlett Avenue, Pittsfield, MA. 01201 413-448-8278

For those of you who would like AMPER-Magic's routines in your own programs, just ask for Anthro-Digital's free-use policy.

AMPER-MAGIC and AMPER-MAGIC COMMAND LIBRARY, Volume 2 are trademarks of Anthro-Digital, Inc.
Applesoft is a trademark of Apple Computers, Inc.

Also available from Anthro-Digital: Versacalc (several versions), Visi-Caids, The Executive Secretary, The Personal Secretary, and Executive Speller, Hebrew II and Hebrew II Plus, S-C Macro Assembler, Quick Trace, Amper-Magic, The Rental Manager, F.A.R.M., The Performance Manager, Omniscan VideoDisc interface, and Flipper. DEALERS INVITED

Circle 41 on Reader Service Card

A Call —A.P.P.L.E. TUTORIAL:

ERROR MESSAGE DICTIONARY

Lee Webster

Introduction

WHEN an error occurs (from BASIC or from the Disk Operating System–DOS), the Applesoft cursor re–appears ( ]), a beep occurs, and a message is displayed.

In "immediate–execution" mode, the form of the message is:

?XX ERROR

In "deferred–execution" mode, the form of the message is:

?XX ERROR IN YY

From the DOS, the form is:

XX

where XX is the name of the specific error, and YY is the line number where the error occurred.

In "immediate–execution" mode and in DOS, the errors are detected immediately. In "deferred–execution" mode, errors are not detected until the statement is executed. "Immediate execution" refers to commands typed from the keyboard, while "deferred execution" means commands that are within the BASIC program, preceded by a line number.

Both the Applesoft Reference Manual and the DOS 3.3 Reference Manual contain information on the error messages. What I have done is to pull this information together in *one* alphabetical list. In this list, the DOS messages are indicated by an asterisk (*) following the listing. If you want further information, I have also included page references for *A—Applesoft BASIC Programming Reference Manual,* and *D—Apple][DOS Manual (3.3).* (A brief listing of Applesoft errors is found in A115–117. A more complete listing of DOS errors is found in D115–122.)

The Error Dictionary

BAD SUBSCRIPT

Attempt to use a variable with a subscript larger than the maximum specified in the DIM statement. Also occurs when calling for a different number of dimensions than specified in the DIM, or when the variable has never been DIMmed and the subscript is greater than 9. (A58,A117)

CAN'T CONTINUE

Attempt to continue a program when no program exists, or after an error, or after a line was deleted or added to a program. (A40,A115)

DISK FULL*

When you try to store information on the disk and storage space is not available. The first thing to do is to: DELETE name, where "name" is the file you tried to store when you got the error. This is necessary as only a partial file has been created on the disk and that can cause problems later. You have two options: delete a file on the disk or save your file on another disk. (D120)

DIVISION BY ZERO

Dividing any number by zero causes an "undefined" result. Be careful to check that the denominators of all fractions are not or cannot be zero. (A115)

END OF DATA*

Occurs when:

1. You try to retrieve data from a disk file when there is nothing there or not enough data. Check your INPUT and GET statements to see how much data they require, and check the way the data file was created to see how much data is stored there.

Causes include: (D117,118)

- too many successive INPUTs or INPUT with too many variables
- too many successive GETs
- byte parameter (B) too large
- relative–field parameter (R) too large in POSITION command
- absolute–field parameter (R) in EXEC command too large (D79,D161)
- record parameter (R) in READ command specified a random–access file record in which nothing has been stored (D86)

2. Attempting to READ after an APPEND without first WRITEing.(D67)
3. Attempting to READ intervening unwritten bytes after WRITEing into bytes beyond last byte of existing sequential text file. (D70)

EXTRA IGNORED

Occurs when:

1. A colon or comma is entered in response to a GET statement.
2. the response to an INPUT statement contains more fields (separated by commas) than the statement expected, or if a colon exists in the expected response (except for a string response). (A67,69)

FILE LOCKED*

Occurs when you try to SAVE, BSAVE, WRITE, RENAME or DELETE a file that has been locked on the diskette. The file will have to be unlocked before you can process these commands. (D35,D120)

FILE NOT FOUND*

Occurs when a DOS command specifies a file name that is not on that particular disk. Only the commands SAVE, BSAVE, INIT and OPEN can refer to a file name that did not previously exist. Check (with CATALOG) to see if the file is there, and if so, check the spelling—you may have misspelled it. (D118)

FILE TYPE MISMATCH*

Occurs when a command attempts to use a file name which has the wrong type for that command. The correct combinations are: (D121)

—◉→

ERROR MESSAGE DICTIONARY

LOAD, RUN, SAVE
— Applesoft, Integer program files
CHAIN
— Integer program files
OPEN, READ, WRITE
APPEND, POSITION,
EXEC
— text files
BLOAD, BRUN, BSAVE
— binary or data files

FORMULA TOO COMPLEX
More than two statements of the form IF "XX" THEN were executed. (A77,A116)

ILLEGAL DIRECT
Cannot use an INPUT, DEF FN, GET or DATA statement in "immediate execution" mode (i.e., without line numbers) (A67, A115)

ILLEGAL QUANTITY
Parameter passed to a math or string manipulation function was out of range, such as: (A115)

- a negative array subscript
- using LOG with a zero or negative argument
- using SQR with a negative argument
- taking the real power of a negative base, i.e., raising a negative number base, i.e., raising a negative number to a decimal power
- null string argument for ASC (A60)
- improper argument for:

function	allowable range
CALL X	X: -65535 to 65535 (A43)
CHR$(X)	X: 0 to 255 (A60)
COLOR = X	X: 0 to 15 (A85)
DRAW N1 AT N2,N3	N1: 0 to 255 N2: 0 to 278 N3: 0 to 191 (A98)
HCOLOR = X	X: 0 to 7 (A89)
HLIN X1,X2 AT Y	X1 & X2: 0 to 39 Y: 0 to 39 (A86)
HIMEM:X	X: -65535 to 65535 (A43)
HPLOT X1,Y1 TO X2,Y2	X1 & X2: 0 TO 279 (A89) Y1 & Y2: 0 to 159
HTAB X	X: 0 to 255 (A50)
IN#N	N: 1 to 7 (A72)
LEFT$(A$,X)	X: 1 to 255 (A60)
LEN(X)	X: 0 to 255 (A59)
MID$(A$,X,Y)	X & Y: 1 to 255 (A61)
ON X GOSUB	X: 0 to 255 (A81)
ON X GOTO	X: 0 to 255 (A81)
PDL(N)	N: 0 to 255 (practically 0 to 3) (A90)
PEEK(X)	X: -65535 to 65535 (A40)
PLOT X,Y	X & Y: 0 to 39 (A85)
POKE(X)	X: -65535 to 65535 (A41)
PR#N	N: 0 to 8 (A72)
RIGHT$(A$,X)	X: 1 to 255 (A61)
ROT = X	X: 0 to 255 (A99)
SCALE = X	X: 1 to 255 (A99)
SCRN(X,Y)	X: 0 to 47 Y: 0 to 47 (A87)
SPC(X)	X: 0 to 255 (A32)
SPEED = X	X: 0 to 255 (A54)
TAB(X)	X: 0 to 255 (A51)
VLIN Y1,Y2 AT X	X: 0 to 39 X: 0 to 39 (A86)
VTAB X	X: 1 to 24 (A50)
WAIT X,...	X: -65535 to 65535 (A41)

I/O ERROR*
Occurs when there is an unsuccessful attempt to store or retrieve data from a disk. Can occur when:

1. Drive door is open—close it. (D26)
2. No disk in drive—put one in. (D26)
3. Disk in drive has not been initialized (prepared for data). (D26)
4. Disk is bad (wrinkled, warped, scratched, etc.). (D26)
5. Disk inserted wrong—take it out and re-insert. (D119)
6. Using a slot parameter naming a slot that doesn't contain a disk controller. (D22)
7. The Verify command finds an error. (D36)
8. Using a drive parameter specifying a drive that doesn't exist on the system. (D119)
9. The system is trying to access a 13-sector disk when using 16-sector DOS. (D119)
10. Diskette is protected or DOS is modified and system has been "booted" with standard DOS.

LANGUAGE NOT AVAILABLE*
Occurs when you try to run a program and the Apple can't find the language (Integer or Applesoft) in either ROM or RAM. (D115)

NEXT WITHOUT FOR
Occurs when:

1. Variable in the NEXT statement does not correspond to the variable in the FOR statement currently in effect. (A79,A116)
2. There is a NEXT statement without any corresponding FOR. (A78,A79,A116)
3. FOR...NEXT loops "cross" each other (improper nesting). (A79)

NO BUFFERS AVAILABLE*
Occurs when the DOS command requires another file buffer and all the available buffers are in use. On booting, three buffers are reserved. All DOS commands except PR#, IN# and MAXFILES require a buffer. (D121,D156)

NOT DIRECT COMMAND*
Occurs when you try to use OPEN, READ, WRITE, APPEND, and POSITION in immediate execution mode. (D122)

OUT OF DATA
A READ statement was executed but all the values in the DATA statements have been used already. Either the program tried to read too much data, or too little data was included in the program. (A70,A116)

OUT OF MEMORY
Occurs when:

1. Program too large. Message sometimes appears when you really aren't out-of-memory, but something (a previously run Hi-Res graphics program, for example) has reset the "pointers" where they shouldn't be for your program. Try turning the Apple OFF (AFTER saving your program to disk). Most specifically, you will get this error:
   a. if HIMEM: sets the highest memory address lower than set by LOMEM: or such that there isn't enough room for the program, or
   b. if LOMEM: is set higher than the current value of HIMEM: or lower than the address occupied by the operating system plus the current program. (A44,A45)

2. Too many variables. (A116)
3. FOR loops nested more than 10 levels deep. (A116)
4. GOSUB's nested more than 25 levels deep. Too complicated an expression. (A80)
5. More than 36 levels of parentheses nested. (A116)
6. When more than 88 dimensions given for an array. That is, DIM A(0,0...0) can have no more than 88 zeros. (A58)
7. Improper exit from a GOSUB (i.e., no RETURN or POP statement).
8. Exiting a FOR...NEXT loop on a conditional statement without setting the index to the specified maximum and without encountering the final NEXT statement.

OVERFLOW ERROR

Calculation result too large to fit BASIC's number format. Range for integer numbers is -32767 to 32767. Range for real numbers is -9.99999999 E+37 to 9.99999999 E+37. (A33) Can also occur for the same reason when using STR$ or VAL functions. (A59,A116)

PROGRAM TOO LARGE*

Occurs when trying to load a disk file into memory and there is not enough memory. This may be related to having HIMEM set too low, and to the language you are using. See the discussion on D122.

RANGE ERROR*

Occurs when parameters in DOS commands exceed allowable ranges. Detailed table in D116:

File Type	Parameter	Range
all	slot	1 to 7
all	drive	1 or 2
all	volume	0 to 254
sequential	byte	0 to 32767
sequential	relative field	0 to 32767
sequential	absolute field	0 to 32767
random	record length	1 to 32767
random	record	0 to 32767
binary	start address	0 to 65635
binary	number of bytes	1 to 32767

DOS Commands:

PR# s	s	0 to 16
IN# s	s	0 to 16
MAXFILES	n	1 to 16

REDIM'D ARRAY

After an array was DIMensioned, another DIM statement for the same array was encountered. (A58,116)

REENTER

Occurs with the INPUT statement if: (A66)

1. Numeric input is not a real, integer, comma or colon.
2. The response is a string containing a quotation mark anywhere within.
3. When user presses RETURN key when a numeric response is expected.
4. An ASCII NUL is the first character in a response.

RETURN WITHOUT GOSUB

A RETURN statement was encountered without a corresponding GOSUB statement. (A80,A116)

STRING TOO LONG ERROR

Attempt made by use of concatenation (putting together) to create a string more than 255 characters long. This would affect the PRINT statement, LEN and VAL functions as well. (A59,A71,A116)

SYNTAX ERROR

Missing parenthesis in an expression, illegal character in a line, incorrect punctuation, etc. (A117) Check the proper SPELLING, FORMAT, and PUNCTUATION of the statement you entered.

Examples:

1. Using ASC function on Ctrl-@. (A60)
2. Attempting to CONTinue after an INPUT has been stopped by Ctrl-C. (A40)
3. DATA statements—cannot use a quotation mark within a string, cannot have a READ statement attempt to assign a DATA element that is a string or a literal to an arithmetic variable. (A69)
4. DEL in the following forms causes this error: (A49)

   DEL
   DEL,
   DEL,
   DEL,—a,b
   DEL a,—b
5. FOR X%...NEXT—cannot use an integer variable for X. Don't let the letter A immediately precede TO, as the computer interprets it wrong:

   FOR I = ERA TO...

   is interpreted

   FOR I = ER AT O...

   and you get the syntax error. (A78,A79)

WHAT ARE YOU WAITING FOR???

Are you tired of waiting for DOS to load and save files? Are you tired of waiting for DOS to finish so you can type again? Are you tired of waiting for your printer? When you buy Diversi-DOS™, you won't have to wait any more! Here's why:

1. DOS speed-up: Apple DOS 3.3 takes 18 disk revolutions to read a single track, whereas Diversi-DOS reads or writes a track in just 2 revolutions. This speeds up file processing tremendously (see table).

2. Keyboard Buffer: Diversi-DOS allows you to type at any time, as fast as you can, without missing a single character.

	APPLE DOS	DIVERSI-DOS
SAVE ‡	27.1 sec.	5.9 sec.
LOAD ‡	19.2 sec.	4.5 sec.
BSAVE*	13.6 sec.	4.1 sec.
BLOAD*	9.5 sec.	2.6 sec.
READ**	42.2 sec.	12.4 sec.
WRITE**	44.6 sec.	14.9 sec.
APPEND**	21.3 sec.	2.3 sec.

* Hi-res screen ‡ 80-sector BASIC program
** 52-sector text file

3. Print Buffer: Diversi-DOS can use a RAM card (16K-128K) to temporarily save characters before they are printed. Thus, your computer won't have to wait for your printer to finish.

4. DDMOVER: Diversi-DOS can now be moved to a RAM card for increased memory with BASIC programs.

Diversi-DOS, the QUADRUPLE utility, requires a 48K Apple II or II+ with DOS 3.3. A simple, menu-driven installation program is included on the un-protected disk. So what are your waiting for?

Send $30 to:

Diversified Software Research, Inc.
5848 Crampton Ct.
Rockford, IL 61111
(815) 877-1343

Visa/Mastercard accepted

Apple is a registered TM of Apple Computer, Inc.

dsr

Circle 42 on Reader Service Card

ERROR MESSAGE DICTIONARY

6. GET—cannot type a RETURN or non-numeric input as a response. (A68)
7. IF...THEN—error if there is THEN without an IF or an IF without a THEN. Also watch out for an A before the T in THEN as discussed above for the FOR. (A76,A77)
8. LIST, Q causes the whole program to be listed, followed by the syntax error message. (A48)
9. Line numbers must be in the range of 0 to 63999. (A35)
10. When RESUME is encountered before an error occurs. (A82)

SYNTAX ERROR*

Occurs when:

1. Syntax error in a DOS command. Check to see that you have the correct format, and correct spelling. Can be the result of an incorrect parameter symbol, missing parameter, incorrect separator (usually need a comma) or if parameter value or command quantity is negative or greater than 65535. (D120)
2. Any field in an EXEC file cannot be interpreted as a valid BASIC command or program line. (D79)

TYPE MISMATCH

Occurs when: (A117)

1. Left-hand side of an assignment statement was a numeric variable and the right-hand side was a string, or vice versa. (A73)
2. Function which expected a string argument was given a numeric one, or vice versa. Can occur when using LEFT$ or RIGHT$ functions too. (A60,A61)
3. The $ is omitted from the command name MID$, so that MID is treated as an arithmetic variable name. (A61)

UNDEF'D FUNCTION

Reference made to user-defined function, but function had never been defined with a DEF FN statement. Can occur if you mistakenly choose a variable name starting with the letters FN. (A74,A117) (A38,A76,A80,A117)

UNDEF'D STATEMENT

Attempt made to GOTO, GOSUB, or THEN to a statement line number which does not exist. (A38,A76,A89,A117)

VOLUME MISMATCH*

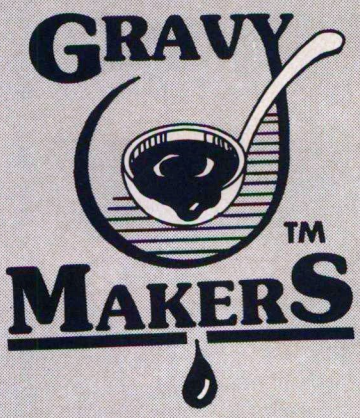
Occurs when the volume parameter in a DOS command is not the same as the volume number placed on the diskette when it was initialized. The volume number from initialization is shown at the top of the CATALOG display. (D118,D119)

WRITE PROTECTED*

Occurs when attempting to store information on disk that has an adhesive label over the cutout on the side, or on a disk with no cutout. (D116,D117)

It's finally here — your computerized cookbook!

Gravymakers brings you the *COOKFLOPPY*™



The *COOKFLOPPY* gives you recipes for six complete meals, including appetizers, soups, salads, breads, vegetables, main courses and desserts — 42 great recipes in all.

Whether you're cooking for one or a crowd of twenty, the *COOKFLOPPY* can adjust recipe quantities to meet your needs, and also provide a shopping list. As a bonus, it can store your favorite recipes for handy future reference. And, if friends admire one of your recipes, it can print them copies!

The *COOKFLOPPY* is so user friendly, a user's guide is unnecessary.

Simply insert the disk in the disk drive, and load the program — your *COOKFLOPPY* does the rest!

Please send me Gravymakers *COOKFLOPPY* Vol. I

Enclosed is my check for $25, ***tax and postage included.***

Name _____

Address _____

City _____ **State** _____ **Zip** _____

Apple II _____ **Apple II+** _____ **Apple III** _____

Mail to: Gravymakers
P.O. Box 367
Auburn, WA 98002-0367

Circle 43 on Reader Service Card

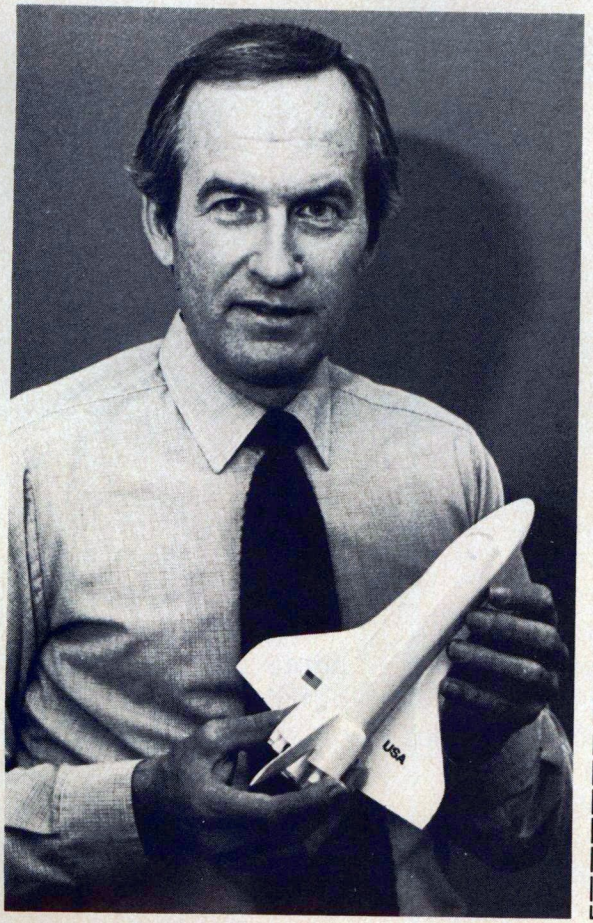
TRACK THE SPACE SHUTTLE ON AN APPLE? YES. WITH MICROSPEED!

At the Jet Propulsion Laboratory in Pasadena, NASA scientists have discovered the power of MicroSPEED. Using this remarkable hardware/software system with an Apple II, they produced a continuous graphic display of the Columbia's position relative to the earth during the second Shuttle mission. This enabled the JPL team to accurately follow the spacecraft in *real time,* and to precisely control its powerful sensors at critical points along the flight path.
Surprised that such a demanding project is possible on the Apple? So were JPL's engineers, and many others who have discovered . . .

THE MICROSPEED DIFFERENCE This extraordinary Language System exploits the real potential of the microcomputer for the first time. The difference between MicroSPEED and other programming languages is that with MicroSPEED, there is virtually *no limit* to what you can achieve. It may well be the ultimate language for the Apple II and III (and soon the IBM Personal Computer). MicroSPEED literally combines the performance of a *minicomputer* with an exhaustive set of user-friendly capabilities: hardware math processing, fast hi-res graphics and text, turtle graphics, print formating, two text editors, unlimited data types, and incredible FORTH extensibility—all at speeds up to 100 times faster than Basic.

USER-FRIENDLY, EASY-TO-LEARN Starting with simple commands that are comfortable even for non-programmers, MicroSPEED extends and builds, allowing you to create your own tailored application languages. The capability of your computer will grow exponentially, as you work in an active partnership with the machine, exploring and developing new problem-solving facilities—creating, correcting, refining your increasingly powerful system.

DEMANDING JOBS AT LOW COST MicroSPEED has been put to the test in fields as diverse as medicine, the stock market, oceanography, and the arts. In even the most challenging applications, MicroSPEED users have been unanimous in their praise of the System and manual. Typical comments are:



". . . we are more than pleased with MicroSPEED . . . I can't imagine using BASIC on any future applications."
Roger Guevremont, National Research Council of Canada.

"I continue to marvel at its versatility and power."
Carl R. Schramm, USCG Base, Kodiak, Alaska.

"Great! . . . A joy to use."
Henry Harris, Mission Design Manager
Jet Propulsion Laboratory

"If you plan to use a personal computer for <u>any</u> demanding task, then we built MicroSPEED for <u>you</u>."
Sam Cottrell, President of Applied Analytics.

MicroSPEED requires the Apple Computer with single disk. MicroSPEED II includes 2 MHz math processor. MicroSPEED II + includes 4 MHz math processor.

Applied Analytics Incorporated
8235 Penn-Randall Place
Upper Marlboro, Maryland 20772 (301) 420-0700

Please send me:

_____MicroSPEED II, $495.00 _____160 Page Manual, $15.00
_____MicroSPEED II +, $645.00 _____Detailed Information

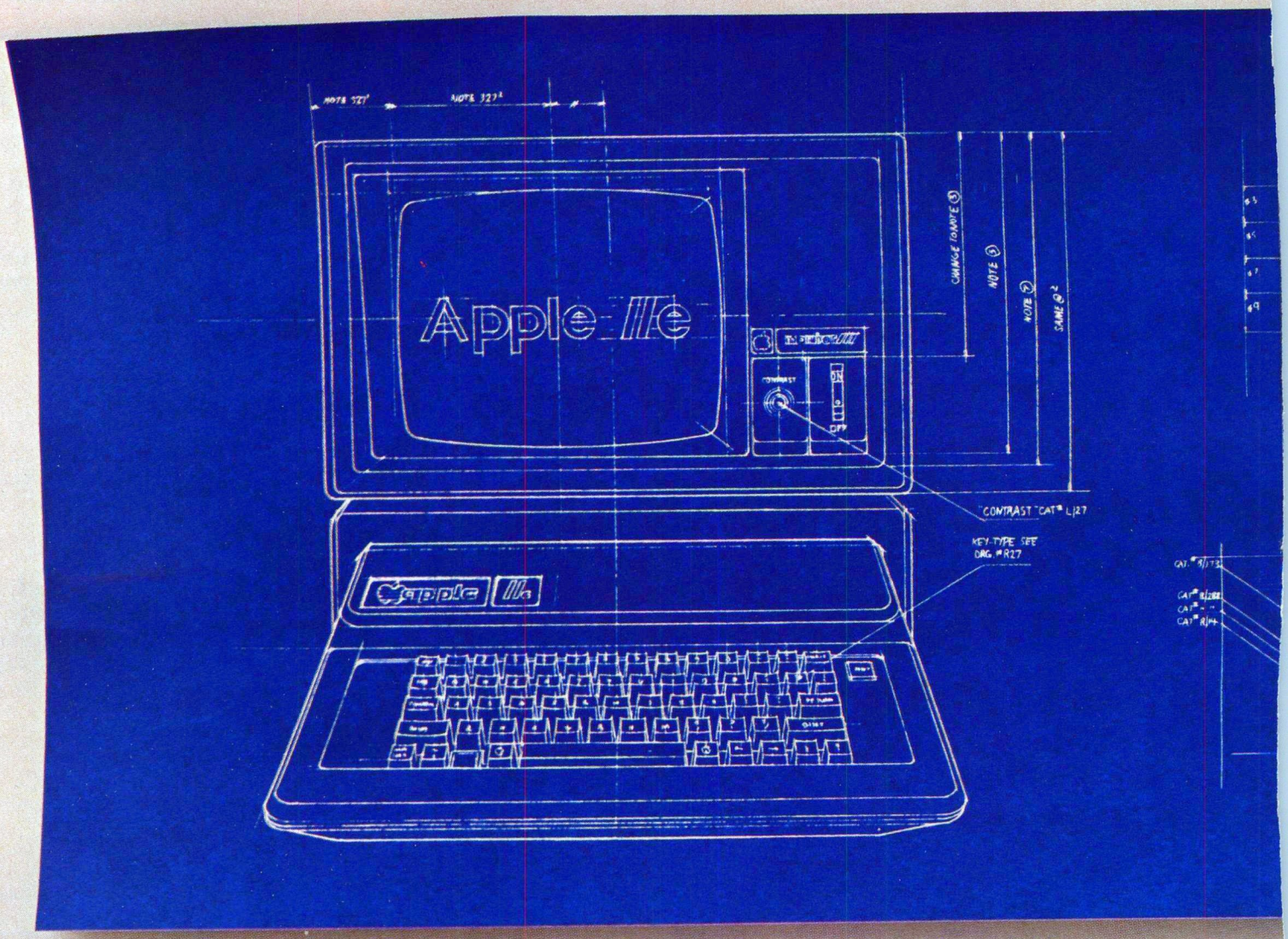
Name: _____
Company: _____
Address: _____
City_____ State_____ Zip_____ Phone No. ( )_____

μSPEED™

APPLE IS A TRADEMARK OF APPLE COMPUTER, INC.

Circle 100 on Reader Service Card

It's the same old Apple II.



For years, people have been trying to build a better Apple® II.

It finally happened.

Meet the Apple IIe, an impressive new version of a most impressive machine.

The "e" means enhanced. Which means a bundle of new features:

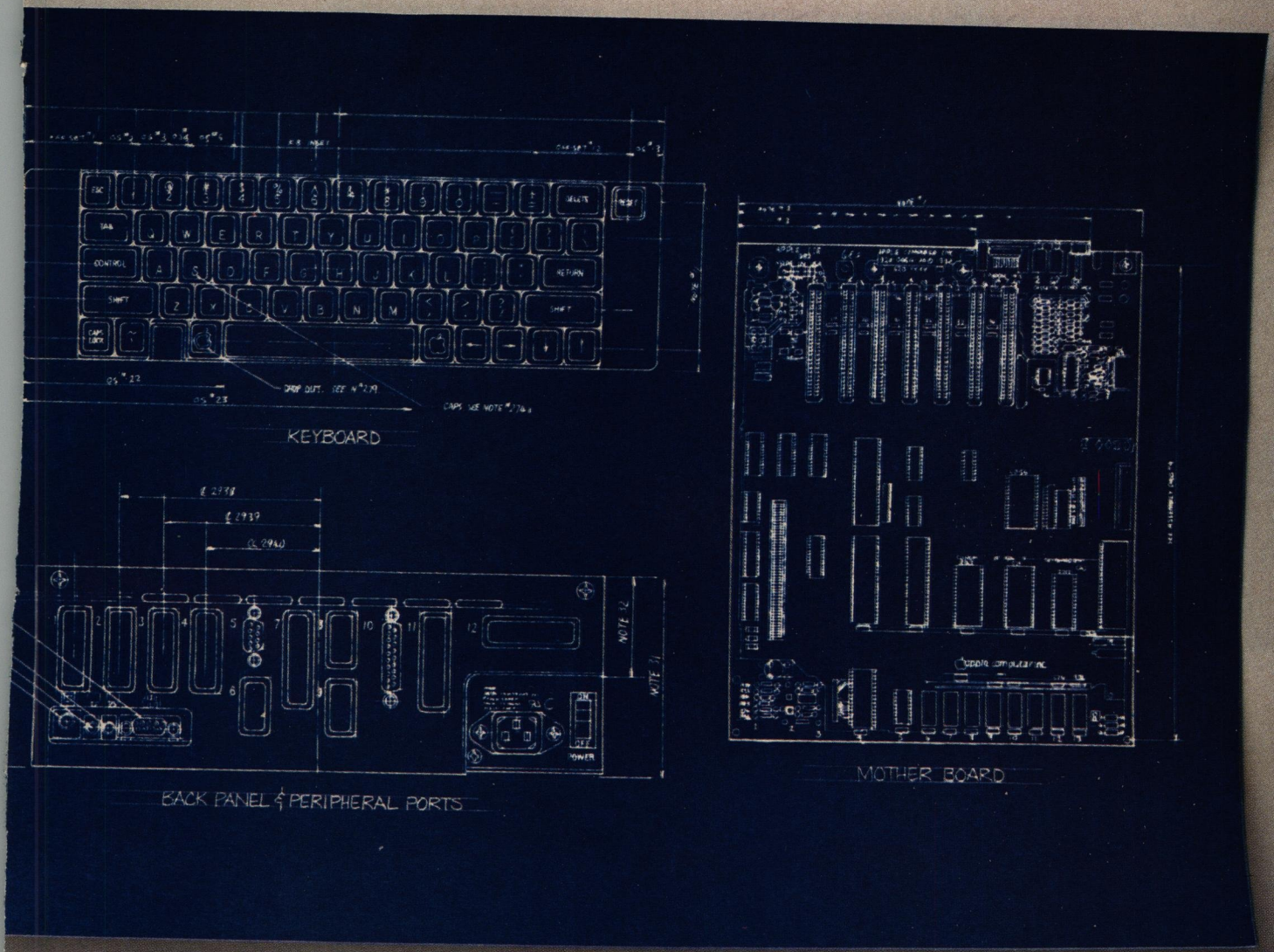
A standard memory of 64K (versus 48K) that's easily expandable. So you can create fatter files and crunch larger numbers of numbers.

A new, improved keyboard, with a complete set of ASCII standard characters. Plus full cursor controls, programmable function keys, and a rapid auto-repeat feature built into every key on the board.

Both upper and lower case characters. (And if you want to see more of them on the screen at one time, a low cost 80-column text card is available.)

Improved peripheral ports. Which make it a lot easier to connect and disconnect game controllers, printers and all those other wonderful things that go with an Apple Personal Computer.

Except for the front, back and inside.



Self-diagnostics. That's a special feature that makes it easy to give your computer a thorough check-up.

Plus an even more reliable design. Achieved by reducing the number of components—which is to say, the number of things that could go wrong.

And bear in mind, the IIe still has all those other virtues that made the Apple II so very popular. Including access to more accessories, peripheral devices and software than any other personal computer you can buy.

So visit any of our over 1300 authorized dealers, and see the newest Apple for yourself.

Like the original, it's rather extraordinary. But then some things never change.

apple®

The most personal computer.

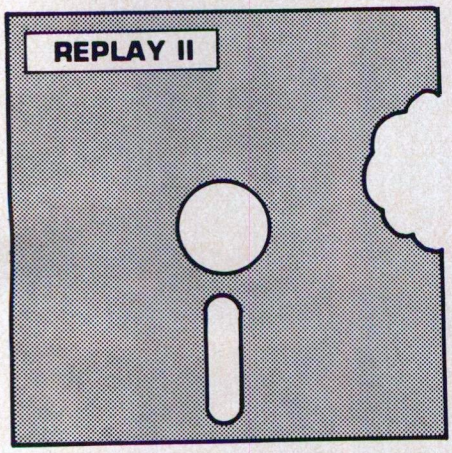
Circle 99 on Reader Service Card

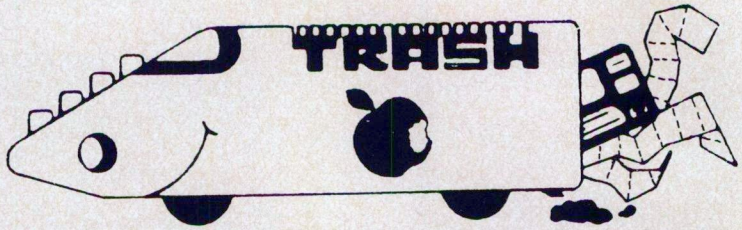
Call **(800) 538-9696** for the location of the authorized Apple dealer nearest you, or for information regarding corporate purchases through our National Account Program. In California **(800) 662-9238.** Or write Apple Computer Inc., Advertising and Promotion Dept., 20525 Mariani Ave., Cupertino, CA 95014. ©1983 Apple Computer Inc.

NEW REPLAY II

APPLE PROGRAM COPY SYSTEM

- **Disk Formatting Irrelevant**
- **Does not interfere with other cards**
- **Menu driven**
- **Card is transparent until copy desired.**
- **Copy *and* restart in under 15 seconds**
- **Will copy all 64k with *no* compression**
- **Copies total load programs**





REPLAY II is an interface card that is slot independent. Users can stop the program, examine and change memory, then restart without changing disks. Control of the system is obtained by pressing the remote switch which comes on an 18 inch cord outside the Apple*. If a copy is desired a blank disk is inserted in Drive 1 and the copy option is selected from the menu. **REPLAY II** does not copy the disk, rather it copies the program executing in memory.

REPLAY II does not change ANY memory - unlike other copy cards. It faithfully reproduces the lower 48k of memory in a fast load format. The upper 16k of memory can also be copied for a 64k copy. Standard DOS 3.3 files can be created for storage on hard disk, although the RAM card is needed.

Utility programs supplied with the **REPLAY II** card include Program Analysis, Packing, and Compression. A language card is not needed to run packed programs. Because most programs are written in Assembly language, the user should be familiar with Assembly in order to fully utilize the Analysis and Packing programs.

NOW game player can save a game at ANY LEVEL and QUICKLY restart it with the **REPLAY II** card.

AT LAST!!! A full sized card with full sized capabilities!!!

TO ORDER OR REQUEST INFO:
Write or Call
Texas Ranch And Shoreline Systems
P.O. Box 15003
Austin, TX 78761

(512) 926-4527

Dealer Inquiries Welcome

COST:
$150.00 (includes postage)
Outside U.S./Canada add $10 shipping
Texas residents add $7.50 sales tax

VISA/MASTERCARD Accepted!!

REPLAY II is intended for archival backup purposes only.

*Apple is a registered trademark for Apple Computers Inc.

Circle 45 on Reader Service Card

A **Call —A.P.P.L.E.** TECHNIQUE:

AN ALTERNATIVE TEXT PAGE

Glen Bredon

ALTHOUGH most Apple][users know that their computer has two text pages, very few have made use of text page two. There are two reasons for this. First, the memory area ($800–$BFF) used by text page two is ordinarily also the location of Applesoft programs. This difficulty is easily overcome since it is, by now, well known how to locate an Applesoft program at any desired location. The second difficulty is that the character output routine (COUT1) in the Monitor does not ordinarily handle page two. Thus no scrolling routine or equivalent exists for page two.

The program presented here provides a simple means for utilizing text page two from Applesoft programs for those people possessing a language card (or other ramcard) and Applesoft on the MOTHERBOARD. It is possible to do similar things on a system without a ramcard, but it would be considerably more complicated than the present simple solution.

There are several reasons why one might like to make use of text page two. You could set up two separate menus on each text page and flip between them instantaneously. Also, you could make use of high resolution graphics page two in the mixed graphics/text mode without any trouble.

Description of the Program Operation

When you assemble TEXT PAGE TWO you will be asked for the value of the label NEWSTART. This will become the starting address for your Applesoft program. You should not specify an address less than $C00 for this. Ordinarily you would want to make this $C00 or, in order to put the program above both Hi-Res pages, $6000.

When the object code is BRUN it will set up the Applesoft &-vector for use of some routines to be described later. It also takes the code from the Monitor ROM and places that on the language card, making the one change that is necessary for the COUT routine to utilize text page two. It then fixes the Applesoft pointers so that your program will be located at the address specified by the value you gave to the label NEWSTART. It also sets up the COUT vector ($36–$37) to point to a special output routine.

The output routine is very simple. It just checks what text page is in use and, if that is page two, uses the monitor on the language card instead of that in ROM. Unfortunately, it is not possible to detect the currently displayed text page by software. Thus we have made use of a memory location to keep track of this. The location used is $E6 which ordinarily holds $20 or $40 in Hi-Res graphics mode according to whether one is on graphics page one or two.

This approach handles most of the problems but not all. The HOME command is done by a monitor routine but it is not done by CHARACTER OUTPUT and hence will always clear text page one regardless of which page is up. To remedy this we have provided an ampersand routine &HOME which clears page 2 (regardless of which page is in view). This works exactly as does the output routine by switching to the monitor on the language card and calling the home routine there.

Three other ampersand commands (&, &&, &NEW) are provided to simplify switching from one page to another.

Using the Program

To use TEXT PAGE TWO you should BRUN TEXT PAGE TWO at the outset from the keyboard or your program should do it from an EXEC file (see below). To switch to page two simply use the command && from your program or the keyboard. To switch to page one, use & alone. The command &NEW will switch from one page to the other, regardless of which is currently in view. Since it is possible to disconnect the output routine by a RESET or other things, we have also provided that the & command (alone) will reconnect this routine.

Use HOME to clear page one and &HOME to clear page two, regardless of which page is in view.

That is all there is to it. You will find that almost everything operates on page two the same as on page one. One exception is the ESCAPE cursor movement. This will work as on page one but a vertical cursor movement on page two will cause the cursor to disappear until a printing character is output. Indeed, this is the best way to determine which page is in use.

It is suggested that your program end with an & in order to assure that page one will be up upon exit from the program. Since the machine language program TEXT PAGE TWO has to reset pointers and therefore does a "NEW," the only way for it to be run from a BASIC program is via an EXEC file. The listing gives an example of this technique and also shows how to use TEXT PAGE TWO. The "PEEK(104)" looks at the high byte of the program address which is ordinarily 8 but is 12 if TEXT PAGE TWO has been BRUN (assuming $C00 was used for the value of NEWSTART during assembly).

LISTING ("DEMO"):

EXEC FIL ("DEMOFILE"):

```
BRUN TEXT PAGE TWO
RUN DEMO
```

```
]LIST
10  IF  PEEK (104) (
    ) 12 THEN  PRINT
    : PRINT  CHR$
    (4)"EXEC DEMOFILE":
    END
20  HGR2 : HCOLOR= 3:
    HPLOT 50,10 TO
    120,10 TO 120,100
30  HPLOT  TO 50,110
    TO 120,110
40  HGR : HPLOT 100,10
    TO 100,110
50  HOME : & : VTAB
    21: PRINT "THIS
    IS TEXT PAGE ONE"
60  &  HOME : &  &  :
    VTAB 23: PRINT
    "THIS IS TEXT
    PAGE TWO"
70  VTAB 22
80  FOR TIME = 1 TO
    20
90  &  NEW : PRINT TIME;
100  FOR PAUSE = 1 TO
    500: NEXT PAUSE
110  NEXT TIME
120  TEXT : & : END
```

ALTERNATE TEXT PAGE

```
                    ********************************
                    *                              *
                    * TEXT PAGE 2 : Glen E. Bredon *
                    *                              *
                    * Call -A.P.P.L.E. : Mar. 1982 *
                    *                              *
                    ********************************

                    NEWSTART KBD                ;New start adrs of program

                    *-------------------------------------
                    * NEWSTART MUST be at least $C00!!!
                    *-------------------------------------

                    CSWL      =     $36
                    PL        =     $E2           ;Used on entry only
                    DOSCLD    =     $3D3
                    CONNECT   =     $3EA
                    AMPER     =     $3F5
                    LOWSCR    =     $C054
                    BASIC     =     $E000
                    PAGHIGH   =     $FBC6
                    VTAB      =     $FC22
                    HOME      =     $FC58
                    COUT1     =     $FDF0

                    * Applesoft stuff:

                    TXTTAB    =     $67           ;Program start +1
                    CHRGET    =     $B1
                    CHRGOT    =     $B7
                    HPAG      =     $E6           ;High byte of HIRES page
                    SCRATCH   =     $D64B

                    * Applesoft token data:

                    HOMETOK   =     151           ;HOME token
                    AMPTOK    =     175           ;& token
                    NEWTOK    =     191           ;NEW token

                    * RAMcard stuff:

                    CONTROL   =     $C080
                    ROMREAD   =     %10
                    ENABLE    =     %11

                              DO    0

                    STADR     MAC
                              LDA   #]1
                              STA   ]2
                              LDA   #>]1
                              STA   ]2+1
                              <<<

                              FIN

                              ORG   $29B

029B: 2C 81 C0                BIT   CONTROL!ROMREAD!ENABLE
029E: 2C 00 E0                BIT   BASIC         ;Applesoft on motherboard?
02A1: 50 57                   BVC   ABORT         ;Exit if not
02A3: AD D5 03                LDA   DOSCLD+2
02A6: C9 9E                   CMP   #$9E          ;DOS on LC?
02A8: B0 10                   BGE   SETAMP        ;Skip if so
02AA: 2C 83 C0                BIT   CONTROL!ENABLE
02AD: A9 00                   LDA   #0            ;Is there a ramcard?
02AF: 8D 00 E0                STA   BASIC         ; (and force reload of card)
02B2: CD 00 E0                CMP   BASIC
02B5: D0 43                   BNE   ABORT         ;Exit if not
02B7: 8D 02 E0                STA   BASIC+2       ;Force BIG MAC reload
02BA: A9 4C         SETAMP    LDA   #$4C          ;Set up & vector
02BC: 8D F5 03                STA   AMPER
                              >>>   STADR.START;AMPER+1
02BF: A9 00                   LDA   #START
02C1: 8D F6 03                STA   AMPER+1
```

```
02C4: A9 03     74              LDA  #>START
02C6: 8D F7 03  74              STA  AMPER+1+1
                74              <<<
                75              >>>  STADR,$F800:PL
02C9: A9 00     75              LDA  #$F800
02CB: 85 E2     75              STA  PL
02CD: A9 F8     75              LDA  #>$F800
02CF: 85 E3     75              STA  PL+1
                75              <<<
02D1: A0 00     76              LDY  #0
02D3: 2C 81 C0  77              BIT  CONTROL!ROMREAD!ENABLE
02D6: B1 E2     78  MOVE        LDA  (PL),Y       ;Move ROM monitor
02D8: 91 E2     79              STA  (PL),Y       ; to RAMCARD
02DA: C8        80              INY
02DB: D0 F9     81              BNE  MOVE
02DD: E6 E3     82              INC  PL+1
02DF: D0 F5     83              BNE  MOVE
02E1: A9 08     84              LDA  #8           ;Change one byte of Monitor
02E3: 8D C6 FB  85              STA  PAGHIGH      ; routine on Ramcard.
02E6: 2C 82 C0  86              BIT  CONTROL!ROMREAD
                87              >>>  STADR,NEWSTART+1:TXTTAB
02E9: A9 01     87              LDA  #NEWSTART+1
02EB: 85 67     87              STA  TXTTAB
02ED: A9 0C     87              LDA  #>NEWSTART+1
02EF: 85 68     87              STA  TXTTAB+1
                87              <<<
02F1: 8C 00 0C  88              STY  NEWSTART     ;Program start must have 0
02F4: 20 4B D6  89              JSR  SCRATCH      ;Sets other pointers
02F7: 4C 1A 03  90              JMP  PAGE1        ;Init page flag & out vector
02FA: 2C 82 C0  91  ABORT       BIT  CONTROL!ROMREAD
02FD: 4C D3 03  92              JMP  DOSCLD
                93
                94  *-------------------------------------
                95  * START must ORG $300 at least!
                96  *-------------------------------------
                97
0300: F0 18     98  START       BEQ  PAGE1        ;& alone
0302: C9 AF     99              CMP  #AMPTOK      ;&&?
0304: F0 23    100              BEQ  PAGE2
0306: C9 BF    101              CMP  #NEWTOK      ;&NEW?
0308: F0 23    102              BEQ  SWITCH
030A: C9 97    103              CMP  #HOMETOK     ;&HOME?
030C: D0 33    104              BNE  BACK         ;Gets syntax error
030E: 2C 80 C0 105              BIT  CONTROL      ;Ramcard on
0311: 20 58 FC 106              JSR  HOME         ;Home text page 2
0314: 2C 82 C0 107              BIT  CONTROL!ROMREAD
0317: 4C B1 00 108  GETNEXT     JMP  CHRGET       ;Back to Applesoft
               109  PAGE1       >>>  STADR,OUTPUT:CSWL
031A: A9 42    109              LDA  #OUTPUT
031C: 85 36    109              STA  CSWL
031E: A9 03    109              LDA  #>OUTPUT
0320: 85 37    109              STA  CSWL+1
               109              <<<
0322: 20 EA 03 110              JSR  CONNECT      ;Connect DOS COUT vectors
0325: A9 40    111              LDA  #$40         ;Fake page 2 for switch
0327: D0 02    112              BNE  SETPAGE      ;Always taken
0329: A9 20    113  PAGE2       LDA  #$20         ;Fake page 1 for switch
032B: 85 E6    114  SETPAGE     STA  HPAG
032D: A2 01    115  SWITCH      LDX  #1           ;Prepare X for page 2 switch
032F: A5 E6    116              LDA  HPAG         ;Assumed to have $20 or $40
0331: 0A       117              ASL               ;$20 -> $40
0332: 10 03    118              BPL  PAGSW
0334: A9 20    119              LDA  #$20         ;$40 -> $20
0336: CA       120              DEX               ;For page 1 switch
0337: 85 E6    121  PAGSW       STA  HPAG
0339: 9D 54 C0 122              STA  LOWSCR,X     ;Switch pages
033C: 20 B7 00 123              JSR  CHRGOT       ;& alone?
033F: D0 D6    124              BNE  GETNEXT      ;Branch if not
0341: 60       125  BACK        RTS
               126
0342: 48       127  OUTPUT      PHA               ;Save character
0343: 24 E6    128              BIT  HPAG         ;Which page are we on?
0345: 50 03    129              BVC  ORD          ;Branch if on page 1
0347: 2C 80 C0 130              BIT  CONTROL      ;Turn ramcard on
034A: 20 22 FC 131  ORD         JSR  VTAB         ;To init correct BASH
034D: 68       132              PLA               ;Retrieve character
034E: 20 F0 FD 133              JSR  COUT1        ;Print it
0351: 2C 82 C0 134              BIT  CONTROL!ROMREAD
0354: 60       135              RTS
0355: B9       136              CHK               ;To check coding

--End assembly--

187 bytes

Errors: 0
```

Organization a problem?

Look what Sensible Software is doing for you...

DISK ORGANIZER

Here is a fast and friendly utility to organize the files on your diskettes using DISK ORGANIZER. You can order the files on a diskette not only alphabetically, but in any order you want, TITLES can be entered, describing a file or group of files on the diskette. These TITLES may be entered in normal, inverse, flashing, and lower case letters. All work is done on a RAM version of the Directory; so if you decide the changes you made are not correct, you can simply abort the process and keep the old catalog. Since DISK ORGANIZER works on a RAM version of the Directory, disk access is kept to a minimum; hence, DISK ORGANIZER is extremely fast.

The following functions are also included:

- DYNAMIC DISPLAY of all file names in the Directory.
- RENAMING the "HELLO" program.
- A SIMULATED CATALOG to show the modified Directory before it is written to the Diskette.
- ALPHABETIZING the file names.
- UNDELETING deleted files.
- PURGING deleted files.
- RENAMING files (with the same character input options as TITLING.
- LOCKING and UNLOCKING (some or all) files.
- DELETING files.
- DELETING DOS for increased data storage.
- A powerful SMART KEY to automatically locate the next valid file name for any specified specified operation.

DISK ORGANIZER...Apple II or Apple II Plus with 48K and one or more Disk Drives...**$30.00**

Is inter-disk travel and intra-disk file manipulation causing difficulties?

Then SUPER DISK COPY is for YOU!

Super Disk Copy is much more than just another copy program. It's practically a 'mini-DOS' in itself. Super Disk Copy is completely menu driven and works with DOS 3.1, 3.2, and 3.3. Files may even be transferred from one DOS to another (example: DOS 3.3 to DOS 3.2). Super Disk Copy makes the conversion to DOS 3.3 less painful. After seeing Super Disk Copy, you will agree it is the best copy program on the market! Among the many features are:

- Wildcards may be entered anytime a file is used. They allow both automatic or user verification of the file handling. (Example: to copy file names that start with 'Apple', you would enter 'Apple=').
- COPY single files (Applesoft, Integer, Text, Binary, Relocatable).
- COPY DOS.
- COPY ENTIRE DISK. Super Disk Copy uses a unique technique for fast copying.
- A BRUTE FORCE COPY made for Pascal, FORTRAN, and CP/M diskettes.
- An optional rearrangement of files so that they occupy contiguous sectors for improved access time.
- UNDELETE deleted files.
- A PLOT of disk usage.
- FIX file sizes. Make sure your valuable files are protected and unused sectors are available.
- DELETE DOS. Frees up extra space on your diskettes.
- ALPHABETIZE file names in directory.
- REPLACE illegal characters in file names.
- Optional 'INIT' of copy diskette.
- LOCK or UNLOCK files and much, MUCH more...

SUPER DISK COPY was one of only two programs to receive a rating of 100 in PRICE/USEFULNESS RATIO in THE BOOK OF APPLE COMPUTER SOFTWARE 1982. They said "This is a definite MUST HAVE utility package."

48K and DISK II required...**$34.95**

For a complete catalog send $1.00, refundable with your first purchase.

If you have a large software collection and have problems locating specific programs or you need to be reminded of what they do...then you need MULTI-DISK CATALOG

Multi-Disk Catalog is designed specifically for keeping track of the contents of the APPLE diskette library. The resulting master catalog can be sorted, searched, and printed. Multi-Disk Catalog is entirely menu driven, easy to use, and very fast. Some of the unique features included are:

- Works with 13 and 16-sector formated diskettes (DOS 3.1, 3.2, and 3.3). Diskettes from either type DOS can be added to the same data-base used.
- You assign the volume identification number (range 1 to 999); there is no need to INIT each of your disks with a unique volume number.
- A 2 letter classification code can be added to each file name. The classifications are retained when the disk volume is updated.
- Multi-key sorting. Up to 3 keys may be selected for sorting the file names. The sort keys include disk number, file name, classification, file type, and file size.
- Multi-Disk Catalog is written entirely in machine language, so it is very fast.
- A powerful search mask can be used to find a file name or groups of file names.
- Multi-Disk Catalog supports special printer interfaces.
- Both sides of a diskette may be entered under the same disk number.
- Titles may be entered for each diskette and much more!

"There is no doubt about it, this is the best and most versatile master catalog program available"—THE BOOK OF APPLE COMPUTER SOFTWARE 1981.

48K and one Disk Drive required...**$25.00**

Sensible Software

6619 Perham Drive • West Bloomfield, Michigan 48033 • Phone (313) 399-8877

Visa and MasterCard welcome, please add $1.25 postage and handling per diskette.

Circle 46 on Reader Service Card

A **Call —A.P.P.L.E.** REVIEW:

APPLE FLASHER

Walter G. Hopkins/Reviewer

APPLE FLASHER *$34.50*
Crow Ridge Associates Inc.
P.O. Box 1
New Scotland, NY 12127
Copy protected *back-up $7.50*

APPLE FLASHER is a graphics package designed to present a series of high-resolution images in much the same way that a slide projector projects images from photographic slides. It is also useful in other display modes to find and display all binary files on a diskette which are interpreted as high-resolution images. According to the literature supplied with the diskette, it would take at least 25 minutes to look at each of 14 images on 10 standard diskettes. The use of APPLE FLASHER reduces that time to about 5 minutes. Since I didn't have 10 diskettes full of binary images of high-resolution screens, I took their word for this claim. The demonstration disk which was in the review package had 15 binary files representing a full complement of high-resolution images and took 30 seconds to load and display all of them.

Before going into detail about the program options, it might be useful to review quickly the capabilities of the Apple II and its high-resolution graphics. As most users are aware, there are two areas of memory which can be converted to a high-resolution image. Page one, which is invoked from Applesoft by the HGR command, starts at memory location $2000 and page two, invoked by the HGR2 command, starts at memory location $4000. Each page is nominally $2000 bytes long, but since a few bytes in this range are not actually displayed, it is only necessary to save $1FF8 bytes instead of the $2000 described in the DOS manual. The saving may seem small, but when the binary file is actually saved to disk, it results in the saving of one sector. Instead of the 34 sectors normally required, only 33 sectors are used. An image may be displayed on either page one or page two by the appropriate BLOAD parameter (A$2000 for page one, A$4000 for page two). Internal switching in the text display flags will then cause the appropriate page to be displayed. A third page of high-resolution data could be stored at $6000 or wherever one desired, but the Apple will not display this page directly. By means of a memory move assuming this data were in "page 3," it could be transferred quickly to page 2 for display. Apparently, APPLE FLASHER uses a technique of this type.

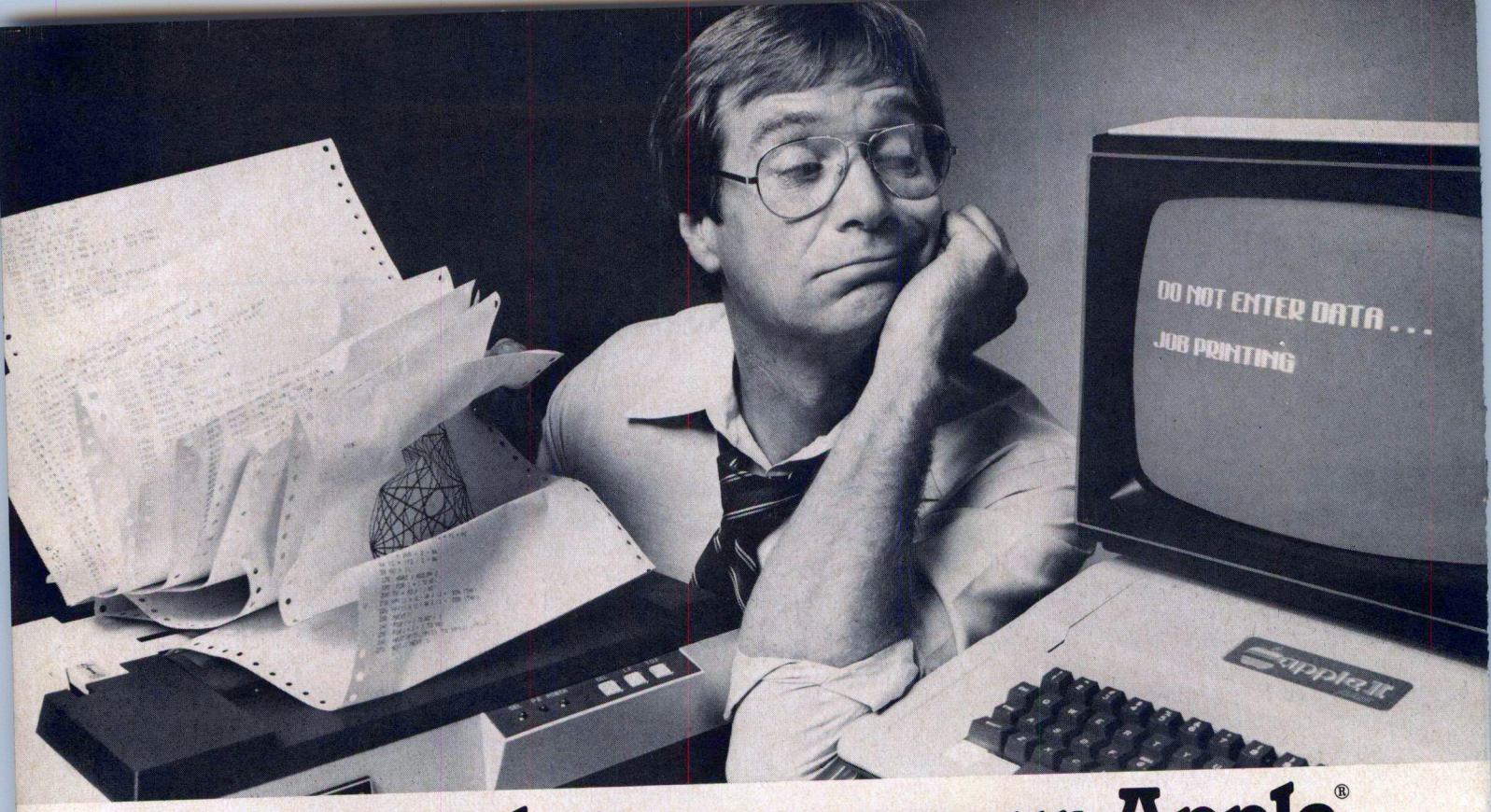
Using APPLE FLASHER is straightforward and almost entirely by menu selections. After booting from the program disk (which is not copyable), a diskette containing binary "pictures" is inserted into the disk drive. If you have two drives, then both may be used to hold "pictures" and may be selected by typing "1" or "2." Depressing any key except "1," "2" or "ESC" will then cause the directory to be scanned and all 33 or 34 sector binary file names will be displayed with a letter "key" preceding each. Pressing the key for such a letter will show the corresponding picture in about 2 seconds. This mode of operation might be useful if you've forgotten the name of a picture and want to quickly scan a few pictures.

If you want to look at all the pictures on a disk, then the "scan" mode may be selected. Typing "S" will automatically show in sequence all the pictures in the most recently read drive. The pictures are shown for about 2 seconds each making this an efficient way to quickly search a disk. The scan may be temporarily interrupted by striking any key except the spacebar. Then any key (excluding the spacebar) will resume the scan. Hitting the spacebar, either during the scan or during a halt, will cause the display to show information about the picture on the bottom section of the picture. A full display is resumed by pressing the spacebar again.

The projector mode is interesting and is entered by typing [Ctrl-P]. In this mode, the Apple acts like a slide projector. At any time, one picture is on display and two more are in memory ready to be displayed by a touch of the arrow keys or by paddle 0 or paddle 1 buttons. If the buttons are used, the position of the potentiometer knob determines whether to move forward or backward in the picture sequence. This allows the disk to hold a slide show and for the user to stand aside with a remote control in the way one might with a slide projector.

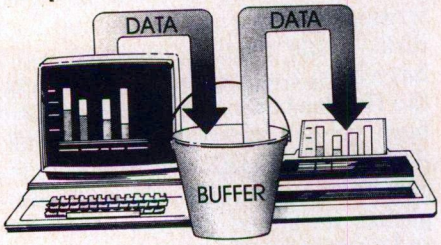
The remaining mode which can be used is the "zip" mode invoked by [Ctrl-Z]. In this mode, the "pictures" are displayed in the order stored on the disk, and for a length of time which is user programmable from 2 seconds to 255 seconds. This is accomplished by first setting memory locations 8312-8319 to zero, and then issuing a POKE into 8314 of the number of seconds while the desired image is being displayed on page one. A BSAVE of the image now contains the necessary timing information (which is displayed by the spacebar depression as mentioned earlier).

The manual is only seven pages in length, but the operation is very straightforward and appropriate choices are easily made by making menu selections. One disadvantage to this program was the inability to store the program itself along with a few pictures on a single disk, thereby allowing a "boot-and-run" type display to be developed. Since a back-up or replacement disk costs $7.50, it may be prudent to avoid writing anything to the original boot disk. If the runtime package could have been transferred to a disk with the Hi-Res binary files, it would have been greatly appreciated. All in all, APPLE FLASHER is a relatively inexpensive program which accomplishes its stated purpose nicely. It had no apparent bugs and was clearly described and documented.



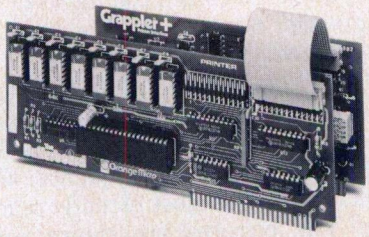
If your printer uses your Apple® more than you do, you need The Bufferboard.™

If your Apple is locked into the "PRINT" mode so much that you've taken up solitaire to kill the boredom, you need a buffer. And if your computer is the Apple II or III, the only buffer for you is The Bufferboard. Expandable to 64K of storage, The Bufferboard stores an instantaneous **bucketful** of print data from your computer. Then it feeds the data to your printer at its own printing rate. Your Apple is set free from driving your printer and is ready for more data from you.



Take your existing interface—and buffer it!

Only The Bufferboard has a simple Interface-Docking System. No bulky boxes or expensive power supplies are needed because The Bufferboard fits right into your Apple—and **docks** onto your existing printer interface. The result is convenient



and economical buffering of most popular printer interfaces, including the Grappler + ™ interface, Epson interface, and Apple printer interface. Thirty seconds and a single hook-up are all you need to end the printer waiting game forever.

Up to 20 letter-size pages stored at a time.

The Bufferboard comes standard with 16K, and is expandable to 32K or 64K of buffering capacity with the addition of memory chips. This "bucket" will hold up to 20 pages of a print job, allowing you freedom to use your Apple.

The Bufferboard—designed exclusively for the Apple Computer.

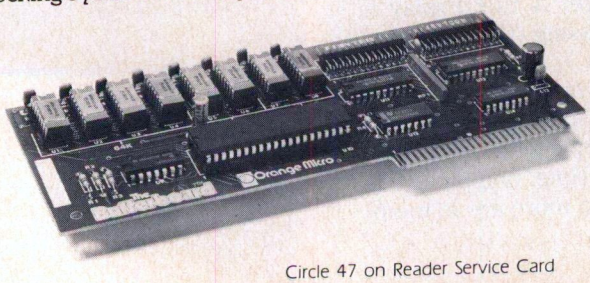
Specifications:

• Versions for Grappler + interface, Epson interface, Apple interface, and other popular printer interfaces • 16K buffer standard • Upgradeable to 32K or 64K • Automatic memory configuration • Automatic self test • Includes interface docking cable.

The Bufferboard is made by Orange Micro, Inc.; the same people who brought you the popular Grappler + printer interface. Both the Grappler + and The Bufferboard are now available at your local Apple dealer.

Apple is a registered trademark of Apple, Inc.
Epson is a registered trademark of Epson America, Inc.

Orange Micro inc.
1400 N. Lakeview, Anaheim, CA 92807
U.S.A. (714) 779-2772
TELEX: TX 183511 CSMA



The Bufferboard™
For Apples and Printers

Circle 47 on Reader Service Card

© Orange Micro, Inc. 1982

IMAGINE IT...

CAPTURE IT.

COMPLETELY REDESIGNED. NOW, THE GRAPPLER + .

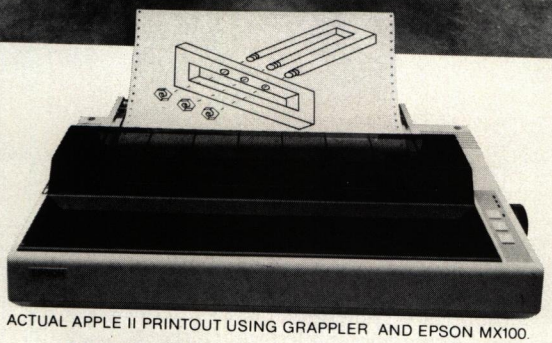
The original Grappler was the first graphics interface to give you hi-res screen dumps from your keyboard. The new Grappler + with *Dual Hi-Res Graphics* adds flexibility with a side-by-side graphics printout of page 1 and page 2.

The Grappler + can now be used with the Apple® Dot Matrix, the Okidata 84, and is Apple III compatible.* In addition, the IDS Grappler + is currently available with color capability, including color graphics screen dumps.

UP TO 64K BUFFER OPTION
An optional Bufferboard can now be added to all existing Grappler and Grappler + interfaces. See your Apple Dealer for details.

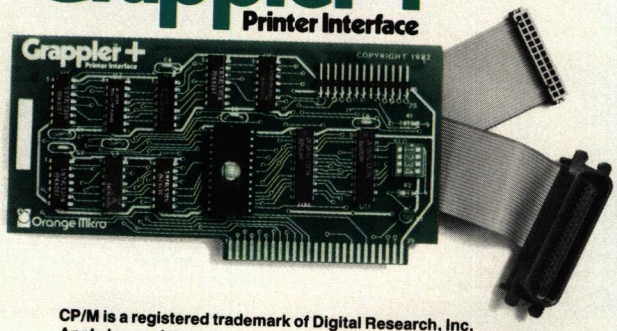
*Requires additional software driver.
**Requires graphics upgrade.

© Orange Micro, Inc. 1982



ACTUAL APPLE II PRINTOUT USING GRAPPLER AND EPSON MX100.

With The Grappler™ + Printer Interface



CP/M is a registered trademark of Digital Research, Inc.
Apple is a registered trademark of Apple Computer, Inc.

THE GRAPPLER + FEATURES:
• Dual Hi-Res Graphics • Printer Selector Dip Switch • Apple III Compatible* • Graphics Screen Dump • Inverse Graphics • Emphasized Graphics • Double Size Picture • 90° Rotation • Center Graphics • Chart Recorder Mode • Block Graphics • Bell Control • Skip-over-perf • Left and Right Margins • Variable Line Length • Text Screen Dumps • also works with Pascal and CP/M®

THE GRAPPLER + INTERFACES WITH THE FOLLOWING PRINTERS:
• Anadex • Apple Dot Matrix • Centronics 122 • C. Itoh ProWriter • DMP-85 • Epson Series • IDS 460, 560, Prism 80 and 132, Microprism • NEC 8023 • Okidata Series** • Star Micronics Series • and many other printers**

Orange Micro inc.

**1400 N. Lakeview Ave.,
Anaheim, CA 92807 U.S.A.
(714) 779-2772 Telex: 183511 CSMA
Foreign Dealer Inquiries Welcome**

Circle 48 on Reader Service Card

CRACK-SHOT™

Internationally acclaimed
The ultimate in copy de-protection

CRACK-SHOT is a total system comprised of hardware and software modules. CRACK-SHOT is easy to use, designed for users and programmers. Eliminates "owners paranoia" about disk failure. Provides quick, easy, reliable archival backups for critical software with a flip of a switch. Allows transferal of software to hard disk. Be an Arcade King. Use CRACK-SHOT as a gaming tool to stop, start and save a game at any level.

**Your total CRACK-SHOT system consists of:
a sophisticated hardware device,
advanced state software utilities,
and a 70-page users manual.**

Price: $149.95

**Optional Hot-line available for only $15.00
24 hours a day, 7 days a week, providing
system updates to registered owners.**

**Cash, check, money order, MasterCard
or VISA accepted.**

**for MasterCard and VISA orders *only*!
Call now! Toll free!
1-800-824-7888 Ask for operator 68
in California
1-800-852-7777 Ask for operator 68**

Pirates Harbor

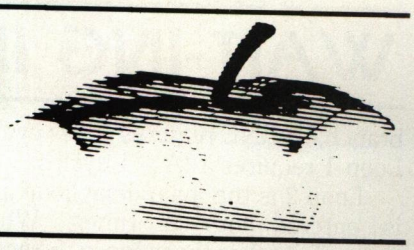
P.O. Box 8928, Boston, MA 02114 617-738-5051 MODEM

Apple is a registered trademark of Apple Computer. CRACK-SHOT is the trademark of Pirates Harbor.
System requirements: 48K Apple II or Apple II Plus, 1 disk drive, Ramcard helpful but not required.
CRACK-SHOT is intended to be used as a programmers aid, gaming tool, and legal archival backup of your programs.
CRACK-SHOT should not be used for illegal purposes.

Circle 49 on Reader Service Card

FROM THE VERY CORE OF APPLE:

WAIT-ING IN THE MONITOR



John K. Elberfeld

BEFORE writing my own machine language timing routine, I attempted to unravel the mysteries of WAIT. The effort was considerable, but with help from some good friends, it was successful and rewarding. Besides, who can look at 1/2 (26+27A+5A*A) without wondering where those numbers came from?

WAIT is a monitor subroutine described in the *Apple Reference Manual* (page 63) as one which "delays for a specific amount of time, then returns to the program which called it. The amount of delay is specified by the contents of the accumulator. With A the contents of the accumulator, the delay is 1/2 (26+27A+5A*A) micro seconds. WAIT returns with the A register zeroed and the X and Y registers undisturbed."

The WAIT subroutine starts at $FCA8 in the monitor-Read Only Memory (ROM). In decimal terms, this is 64680 or -856. Page 147 in the Apple Reference Manual provides the following information –

FCA8: 38	WAIT	SEC	2
FCA9: 48	WAIT2	PHA	3
FCAA: E9 01	WAIT3	SBC #$01	2
FCAC: D0 FC		BNE WAIT3	2/3
FCAE: 68		PLA	4
FCAF: E9 01		SBC #$01	2
FCB1: D0 F6		BNE WAIT2	2/3
FCB3: 60		RTS	6

The first column of figures gives the exact hexadecimal address locations in ROM where these commands are stored. The second column is the hex representation of the numerical format of the 6502 processor machine language commands. Since these numbers are confusing and meaningless to humans, the same commands are repeated in columns three and four in human readable form. The third column gives an English name to a particular memory location (WAIT2 represents location FCA9, for example). The fourth column is the mnemonic or assembly language form of the instruction. Because each mnemonic can represent several distinct machine language commands (depending on numbers or addresses used) the exact number of the machine operation code in column two is important in determining the time required for each command.

The fifth column is *not* listed in the manual. It represents the number of clock cycles needed by the 6502 processor in the Apple to execute each command. The number of cycles required for each command is listed in many 6502 Assembly Language text books. If we know the number of cycles required in a program, and the length of each cycle, we can compute the time required for program execution. Notice that BNE requires three clock cycles if the branch is successful, but only two clock cycles if the branch is *not* successful.

The flow chart on the next page represents the program operation —

The next chart traces the number of clock cycles required as the program is executed. The demonstration assumes the accumulator is loaded with 4.

JSR	6										
SEC	2										
PHA	3				3			3			3
SBC	2	2	2	2	2	2	2	2	2	2	2
BNE	3	3	3	2	3	3	2	3	2	2	2
PLA				4			4		4	4	4
SBC				2			2		2	2	2
BNE				3			3		3	2	2
RTS											6

The total number of clock cycles is 107. Using the formula:

1/2 (26 + 27 A + 5 A*A)
with A = 4

This also results in 107 cycles. This gives proof that the program is understood correctly. The challenge now is to compute the number of clock cycles needed when a general number A is loaded into the accumulator.

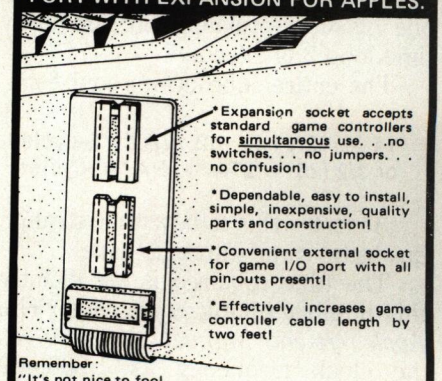
The initial commands of JSR and SEC, and the final command of RTS are used only once so they account for 6 + 2 + 6 = 14 cycles.

Loop 1 is designed to decrease the value of A by 1 each time it is executed. It gets fancy with the stack to preserve the values in the X and Y registers. Specifically, the starting value of the accumulator is stored in the stack. This value now stored in the accumulator and the stack controls how many times Loop 2 is executed, and the value in the accumulator is changed in the process. When Loop 2 is finished, the starting value of A is taken from the stack and pushed back into the accumulator. The value is then decreased by 1, and this new, smaller value is pushed onto the stack for storage. Loop 2 now executes a smaller number of times, and Loop 1 continues to decrease A until zero results. When 0 is the result of the subtraction, the program does *not* branch back, but returns to the calling program.

Loop 1 uses PHA, PLA, SBC, and BNE for a total of 3 + 4 + 2 + 3 = 12 cycles. It will execute A times, and on the last execution there is no

ProLine X-10

EXTERNALLY ACCESSABLE GAME PORT WITH EXPANSION FOR APPLES.



*Expansion socket accepts standard game controllers for simultaneous use. . .no switches. . . no jumpers. . . no confusion!

*Dependable, easy to install, simple, inexpensive, quality parts and construction!

*Convenient external socket for game I/O port with all pin-outs present!

*Effectively increases game controller cable length by two feet!

Remember:
"It's not nice to fool with the mother board!"

See your dealer or order direct. . .Add $1.00 handling on direct orders. Check, Visa or Master Card is acceptable.

ProLine X-10 Only $19.95

Pro Line Electronics

Box 735, Higginsville, MO 64037

Apple is a registered trademark of Apple Computers, Inc.

Circle 50 on Reader Service Card

WAIT-ING IN THE MONITOR

branch, so BNE requires 1 less cycle. Loop 1 requires 12A – 1 cycles.

Loop 2 is the main delay loop and is entered into A times. When entered, it takes the value in the accumulator and decreases it by 1. If the result is *not* zero, it loops back and repeats this process until zero does result. The first time through it loops A times, but when the program passes through loop 2, the value of A is decreased by 1 so on the next execution, loop 2 only goes through the decreasing process A–1 times. The total number of times loop 2 is executed is:

A + (A-1) + (A-2) + ...
3 + 2 + 1

Some high school math shows this adds up to:

A(A-1)/2

Loop 2 uses SBC and BNE for a total of 2 + 3 = 5 cycles each time the branch is successful. It uses only 4 cycles if the branch is *not* successful (which happens each time the value in the accumulator is reduced to 0). The total number of clock cycles is:

1/2 A(A-1)*5 - A

The last term (–A) accounts for one unsuccessful branch for each time loop 2 is executed.

The entire subroutine requires:

14 + 12A-1 + 1/2 A(A-1)*5 cycles
or 1/2 (26 + 27A + 5 A*A) cycles

At last – this is how that strange quadratic came into existence!

The time required is *not* exactly this number of micro seconds. The *Apple reference Manual* (page 3) gives the clock frequency as 1,023,000 machine cycles per second. The time for 1 cycle then is:

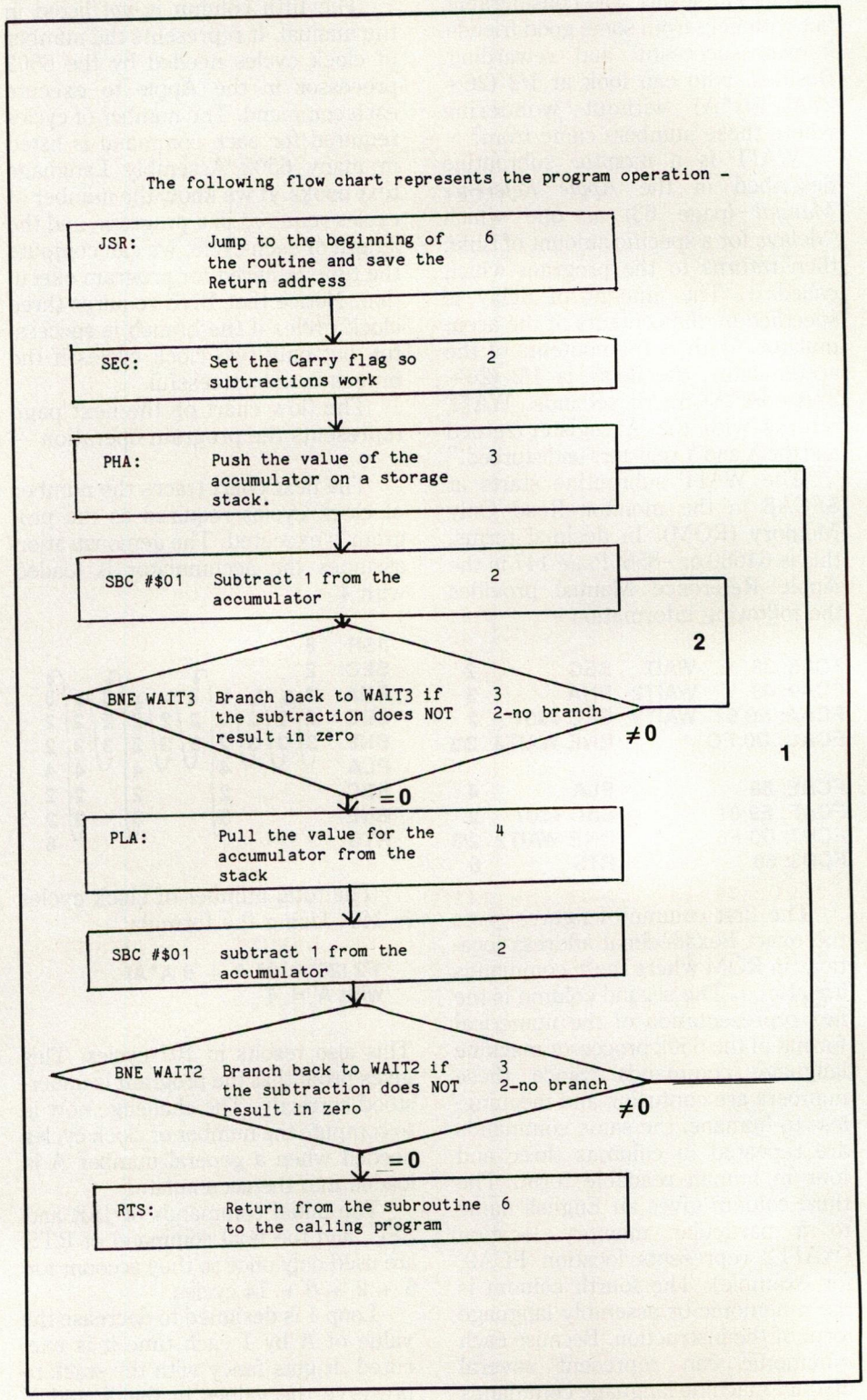
1/1023000 seconds or 1/1.023 micro seconds

Thus a jump to WAIT (including the Jump time) is:

1/2 (26 + 27A + 5 A*A)/1.023 micro seconds

Note: The WAIT routine is also described on page 85 of the *Apple Monitors Peeled*, but the authors incorrectly *multiply* by 1.023 rather than divide by this number. This error is accentuated in the chart on page 86.

My goal for this article is to give you a push to look beyond BASIC. I tried to present a systematic approach to use when examining other routines. My final advice is to read the large print, small print, and the footnotes – but don't believe them just because an "expert" put them down on paper.



WAITING FOR A KEYPRESS

Ken Kashmarek

IN the November 1982 issue, the A.P.P.L.E. Doctor provided a BASIC routine which would approximate a timed input. However, the timing is less accurate than what can be obtained on an Apple. The following routine:

```
:ASM

                    ********************************
                    *                              *
                    *    KEYBOARD WAIT ROUTINE     *
                    *      by Ken Kashmarek        *
                    *                              *
                    * Call -A.P.P.L.E. : MARCH1983 *
                    *                              *
                    ********************************
                    *                              *
                    * USAGE: 100 &WAIT FOR T,R     *
                    *  where T=maximum number of   *
                    *  10ths of a second to wait   *
                    *  for input and must be a     *
                    *  variable. If a key is       *
                    *  pressed, then time left is  *
                    *  returned to R.              *
                    *                              *
                    *     100 T=100                *
                    *     110 &WAIT FOR T,R        *
                    *     120 PRINT "R="; R        *
                    *                              *
                    ********************************

                    *
                    CHRGET   =    $B1
                    KEYBD    =    $C000
                    CHKCOM   =    $DEBE
                    SYNTAX   =    $DEC9
                    PTRGET   =    $DFE3
                    SNGFLT   =    $E301
                    GETBYT   =    $E6F8
                    MOVEMF   =    $EB2B
                    WAIT     =    $FCA8
                    *
                              ORG  $300
                    *

0300: C9 B5         WAIT1    CMP  #$B5       ;Wait?
0302: D0 2C                  BNE  NO
0304: 20 B1 00               JSR  CHRGET     ;CHRGET->A
0307: C9 81                  CMP  #$81       ;FOR?
0309: D0 25                  BNE  NO
030B: 20 B1 00               JSR  CHRGET
030E: 20 F8 E6               JSR  GETBYT     ;GETBYT->X
0311: AD 00 C0      WAIT2    LDA  KEYBD      ;Keypress?
0314: C9 80                  CMP  #$80
0316: B0 08                  BCS  WAIT3      ;Branch if yes
0318: A9 C4                  LDA  #196       ;0.1 sec.
031A: 20 A8 FC               JSR  WAIT       ;Mon WAIT
031D: CA                     DEX             ;X=X-1
031E: D0 F1                  BNE  WAIT2      ;Branch if non-zero
0320: 8A            WAIT3    TXA             ;X->A
0321: A8                     TAY             ;A->Y
0322: 20 01 E3               JSR  SNGFLT     ;Y->FAC
0325: 20 BE DE               JSR  CHKCOM
0328: 20 E3 DF               JSR  PTRGET     --End assembly--
032B: AA                     TAX
032C: 20 2B EB               JSR  MOVEMF     51 bytes
032F: 60                     RTS
0330: 4C C9 DE      NO       JMP  SYNTAX     Errors: 0
```

& WAIT FOR X,Y

uses the wait routine in the monitor. The user specifies a maximum time value in X. If a key is pressed before this time is up, then control is returned with the residual time in Y. If no key is pressed, then control is returned with Y=0. The time value is in units of tenths of a second (X=100 is 10 seconds). The keyboard is checked every 0.1 seconds for a keypress. &WAIT FOR is used as follows:

```
100  X = 100:REM set 10
     seconds max
110  PRINT "user prompt ";
120  & WAIT FOR X,Y:REM wait
     for key
130  IF Y = 0 THEN GOTO 200:
     REM timeout
140  INPUT "";A$REM else read
     input
150  PRINT "time = ";(X-Y)/10
```

See if you can get a keypress delay of zero on the sample program. Good Luck!

```
]LIST

100  REM

  &WAIT FOR KEY PRESS DEMO
     BY KEN KASHMAREK

CALL -A.P.P.L.E. : Mar. 1983

110  REM  BY KEN KASHMAREK
115  PRINT "LOADING ROUTINE": GOSUB
     320
120 X = 100: REM  10 SECONDS
130  PRINT "ENTER INPUT ";: REM
     PROMPT
140  &  WAIT  FOR X,Y: REM  START
      TIMER
150  IF Y = 0 THEN 190: REM CHECK
      TIMER
160  INPUT "";A$: REM  READ INPUT
165  PRINT "INPUT=";A$
170  PRINT "TIME USED=";(X - Y) /
     10;" SECONDS"
180  IF A$ = "" THEN  END
185  GOTO 130
190  PRINT " -) TIME OUT ";X / 10
     ;" SECONDS"
200  GOTO 130
320 A$ = "3F5:4C 00 03 N 0300:C9
     B5 D0 2C 20 B1 00 C9 81 D0 2
     5 20 B1 00 20 F8 E6 AD 00 C0
      C9 80 B0 08 A9 C4 20 A8 FC
     CA D0 F1 8A A8 20 01 E3 20 B
     E DE 20 E3 DF AA 20 2B EB 60
      4C C9 DE ND9C6G"
330  FOR I = 1 TO  LEN (A$): POKE
     511 + I, ASC ( MID$ (A$,I,1)
     ) + 128: NEXT : POKE 72,0: CALL
      - 144
340  RETURN
```

MICROSOLITUDE

Mike Shadick

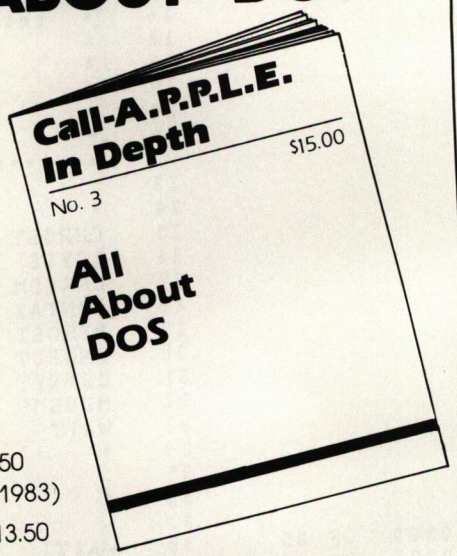
When you want not to be alone
— Nor be with others, either —
You needn't make a compromise.
You can, indeed, choose **neither!**
I call it, microsolitude.
The best of both worlds, this:
Just you and your computer, working
Side by side. Pure bliss!
You feel no social pressures. For
You are, indeed, alone.
Your privacy is quite complete.
You're fully on your own!
Yet are you by yourself? You're **not!**
You're interacting with
A digital companion which **responds**
— And that's no myth!
And yet, the ways it can respond
Are always up to you.
Control of this "relationship"
Is in your hands? How true!
The best of ev'rything, is this.
At least, that's how I see it.
While many things "lock up" one's soul,
Computing helps you **free** *it!*

ANNOUNCING!

A.P.P.L.E. Member Introduction

Price: **$12.50** (offer expires April 30, 1983)

ALL ABOUT DOS



More than 250 pages of DOS tutorials and utility programs.

INTRO PRICE: $12.50
(expires April 30, 1983)
Member price: $13.50
Non-Member price: $15.00
(Available at your local computer store)
Dealer Inquiries Invited

A.P.P.L.E. ORDERS

21246 - 68th Ave. S.
Kent, WA 98032
(206) 872-2245

Wash. St. Residents add 6.8% sales tax.
Foreign Members add $6.00 shipping charge.
Canadian Members, postage being revised.

PASCAL GRAPHICS:

SEEING DOUBLE SAFELY

Lawrence O'Glasser

RON DEGROAT made an excellent find when he discovered how to use the Pascal Turtle Graphics routines to access the Apple's second high resolution graphics page (*Call -A.P.P.L.E.* Volume IV, Number 2, February, 1981). However, there is a potential memory allocation problem when using his technique. Certain conditions will lead to data and code being written to the memory that constitutes Hi-Res page 2. This results in clobbering both the screen (for sure) and the program (probably). This article explains both the problem and the suggested solution.

Under the Pascal system, the bulk of the Apple's memory is divided into three areas (refer to the *Apple Pascal Operating System Reference Manual* (version 1.1), Apple Product #A2L0028 (030-0100-00)):

1. *The Program Stack.*
   This starts at about 44K and grows down (toward lower memory addresses). The low-memory end of the stack is pointed to by KP, the stack pointer.
2. *The Data Heap.*
   This starts around 3K and grows up. The high-memory end of the heap is pointed to by NP, the heap pointer.
3. *Free Memory.*
   Whatever is between NP and KP is available for use by either the program stack or the data heap, as needed.

When KP is less than (or equal to?) NP, all available memory has been exhausted, so the program aborts. Now, when Apple invented the Turtle Graphics package, they needed a way to prevent the program stack from trashing the High Resolution screen (page one), since that chunk of memory is basically right in the middle of "free memory". What they did was to jump the heap pointer NP from 3K up to 16K. This way if the stack grows down far enough to threaten Hi-Res page 1, it bumps into NP and stops.

Thus, all we have to do to give page 2 the same protection which Apple gives page 1 is to move NP up to 24K, the high end of the second High Resolution screen.

Fortunately, the Apple Pascal procedures "new" and "mark" provide access to the heap pointer. "New" allocates dynamic memory to the heap, resulting in the heap pointer being moved. In the routine of Listing #1, "new" is used to allocate Hi-Res page 2 to our program's use. The code appears in the initialization section of the "cDouble" library module, so that it executes before any code in the main program which USES it.

Oddly enough, Apple Pascal executes the initialization sections of library modules in the reverse order from that in which they appear in the USES statement.

In order to use the complete module, do the following:

1. Assemble the routines from DeGroat's article into "CDOUBLE.A.CODE".
2. Compile the module in Listing #1 into "CDOUBLE.P.CODE".
3. Run the linker, specifying host file "CDOUBLE.P.CODE," lib file "CDOUBLE.A.CODE" and output file "CDOUBLE.CODE".
4. Use the Library program to install "CDOUBLE" into "SYSTEM.LIBRARY".

With this addition, Apple Pascal provides a tremendously powerful development system for high resolution graphics. Arise, fellow programmers, and exploit it!

Listing #1

```
(*$S + *)                          (* necessary for UNITs *)
Unit cDouble; Intrinsic Code 23 Data 24;
Interface
    procedure drawScreen(P:integer);
    procedure displayScreen(P:integer);
Implementation
    type
      block8K = packed array [1..8192] of 0..255;
      point8K = ^block8K;
      magicPointer = record case Boolean of
                       true: (pointer: ^integer);
                       false: (value: integer);
                     end;
    var
      hiRes2: point8K;
      heapPointer: magicPointer;

    procedure drawScreen;    External;
    procedure displayScreen ;External;

BEGIN
new(hiRes2);                       (* make hires p2 part of
                                   heap*)
mark(heapPointer.pointer);         (*see if heap pointer is at
                                   *)
                                   (* high end of hires p2*)
if heapPointer.value < > 24576 then
  begin
  writeln('Run-Time error from library');
  writeln('module "cDouble." In your');
  writeln('USES statement, "cDouble" ');
  writeln('must be specified *BEFORE*');
  writeln(' "turtleGraphics. (',heapPointer.value,')');
  exit(program)
  end
END.
```

Don't Pick Just Any Card

Pick the APPLI-CARD™ and APPLI-DISC™ combination and be a Sure Winner!

If you have an Apple® computer, we can deal you a once-in-a-lifetime hand that just can't be beat.

Now you can plug our APPLI-CARD into your Apple and have total CP/M capability. Adding the APPLI-DISC you'll have the potential to speed overall program execution by greater than 10 times while increasing your data storage by more than 30%.

We give you a choice of either a 64K or 128K ram extender. Add this to the APPLI-CARD's existing 64K and you have a 128K or 192K APPLI-DISC. Eliminate the need for an expensive add-on disk drive and end up with a faster system with more storage and more memory for a lot less money!

We've dealt you an unbeatable combination. Now it's up to you to call and cash in on a great deal. Ask about our 8088 Card, another innovative product from PCPI.

4 MHz 64K APPLI-CARD w/APPLI-DISC Software	$295
4 MHz 64K APPLI-CARD w/ 64K Ram Extender / APPLI-DISC Software	$395
6 MHz 64K APPLI-CARD w/APPLI-DISC Software	$375
6 MHz 64K APPLI-CARD w/ 64K Ram Extender / APPLI-DISC Software	$495

PCPI
Personal Computer Products, Inc.

THE COPROCESSING CO.
16776 Bernardo Center Drive
San Diego, CA 92128
(619) 485-8411 Circle 53 on Reader Service Card

APPLI-CARD and APPLI-DISC are trademarks of PCPI.
Z-80 is a registered trademark of Zilog, Inc.
CP/M is a registered trademark of Digital Research, Inc.
WordStar is a registered trademark of Micropro, Inc.
dBase II is a registered trademark of Ashton-Tate.

Prices subject to change without notice.

MORE INFORMATION about the products and services in this issue FREE!

Just circle the number and fill out the card. (Note: card expires three months following month of publication.)

READER SERVICE CARD

Free information on the items circled

Card Expires May 31, 1983 (3-83)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180
181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210

1. What kind of product(s) are you likely to buy in the next three months?

A. ☐ Hard Disk
B. ☐ Increased Capability Floppy Disk
C. ☐ Apple Compatible Floppy Disk
D. ☐ Computer
E. ☐ Printer
F. ☐ Modem
G. ☐ CRT

2. More information on Apple PugetSound Program Library Exchange? A. ☐ Yes B. ☐ No

Name _____

Company _____

Address _____ City _____

State _____ Zip _____ Phone _____

Join!

A.P.P.L.E.

Apple PugetSound Program Library Exchange

THE **WORLD'S** LARGEST COMPUTER USER GROUP

Be a part of the over 16,000 A.P.P.L.E. members and you'll receive:

- 12 ISSUES OF **Call —A.P.P.L.E.** Magazine
  The nation's premier Apple computer publication.
- Substantially reduced prices on sophisticated software, hardware, publications, and services.
- Access to the A.P.P.L.E. closed user group on the **SOURCE**. Excellent for information exchange and program downloading.
- **Call —A.P.P.L.E.** Hotline.

ALL THESE BENEFITS AND MORE FOR ONLY $45!

Act Now!

☐ Please send me an A.P.P.L.E. Apple-cation and information.
☐ $45.00 payment enclosed.
☐ Mastercharge/Visa

$25.00 First time fee
20.00 First year dues
$45.00 Total dues

MasterCard # _____ Exp. Date _____

VISA # _____ Exp. Date _____

Name _____

Address _____

City _____ State _____

Zip _____ Phone _____

A.P.P.L.E., 21246 68th Ave. S., Kent, WA 98032
(206) 872-2245

First Class mail for **Call — A.P.P.L.E.** add $15. Otherwise third class mailed free as part of membership.

Additional postage required for foreign memberships and shipments.

Additional membership postage:

- Canada, West Indies, Central America, Columbia, Venezuela and Mexico — Add $15.00 *($60.00 total)* ***Air Mail***
- All S. American countries (except above), Europe, and N. Africa — Add $21.00 ($66.00 total)
- Asia, Mid East, S. Africa, Australia and Pacific Islands — Add $27.00 ($72.00 total)

All orders must be prepaid in U.S. funds only.

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 217 CLINTON, IA

POSTAGE WILL BE PAID BY ADDRESSEE

Call —A.P.P.L.E.
P.O. Box 1993
Clinton, IA 52735

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 32 KENT, WA

POSTAGE WILL BE PAID BY ADDRESSEE

A.P.P.L.E.
21246 68th Ave. S.
Kent, WA 98032

A **Call —A.P.P.L.E.** UTILITY:

CP/MUFFIN

Val J. Golding

CP/MUFFIN was a rush job. We had a story submitted to us on Wordstar on a CP/M formatted diskette. Fortunately for us, we had just published a story in the December *Call -A.P.P.L.E.* by Art Messler, which pretty well explained the structure of a CP/M directory. As it turned out, the clues in that story were sufficient to start us on the right path. We were working in the dark, not having a Z-80 system ourselves, thus writing a program to convert CP/M files to standard DOS presented an intriguing challenge.

At the same time, we had just finished editing a story which was published in "All About DOS" by Bill Parker, which outlined some simple techniques to call RWTS from BASIC. The basic premise of Bill's routine was to repoint two Applesoft strings at pre-existing data in the RWTS data buffer and write them to disk.

With Bill's short assembly program as a core, we set about getting the data from the CP/M disk into a form we could interpret. In brief, this meant reading the CP/M directory into an array, checking for and bypassing deleted and dummy files, then presenting the user with the option of copying the entire disk or single files. Later, after having done some test conversions, we determined it would be helpful to have a choice of removing control characters from the file or removing [Ctrl-J] only. (CP/M uses a [Ctrl-J] after each carriage return.)

CP/M data is stored on a diskette in manner different than standard DOS, with respect to the sequence in which sectors are read, thus based on Messler's article, we had to establish an array to be used as a look-up table, then base our read on the four block patterns in the table. This was the essence of the whole program, since DOS RWTS can read any sector on a CP/M or Pascal diskette.

We put this together over a weekend and ended up with a working program, albeit slow. (45 minutes to convert a single diskette is not exactly the world's land speed record.) At this point, having gotten our needed results, we called in David Sparks, who waved his magic wand, added an additional assembly language section which laid out a buffer to handle an entire CP/M file and *then* write it to disk.

The BASIC program is well modularized and with the REMs should be easy to follow. It will now convert a full CP/M disk in about seven or eight minutes. This speed could be improved on by deleteing line 1030 which puts a MONO into effect. CP/M does not use a track/sector list to keep track of its files; this is done via directory entries and dynamic block allocation, therefore it is possible although unlikely to encounter a DISK FULL ERROR while converting a very full CP/M diskette. The BASIC program is complete in itself, since the machine code (which is fully relocatable) is loaded using the S. H. Lam routine. The assembly listing, which is well commented, is for the use of those who wish to further expand on the program. It should be fairly simple to write a program to convert DOS files to CP/M.

```
                    ********************************
                    *                              *
                    *          CPMSTUFF            *
                    *                              *
                    *  A colaboration between Bill *
                    *   Parker, Val Golding and    *
                    * David Sparks, who tied up    *
                    * all the loose knots...       *
                    *                              *
                    * Call -A.P.P.L.E. March 1983  *
                    *                              *
                    ********************************

                    LINNUM    =    $50
                    BYTPTR    =    $50
                    FRESPC    =    $71
                    VARPNT    =    $83
                    LSTBYTHI  =    $A0
                    LSTBYTLO  =    $A1
                    STRNG1    =    $AB
                    CHRGET    =    $B1
                    AMVEC     =    $3F5
                    BUF1      =    $B4BB
                    BUF2      =    $B53B
                    DATA      =    $D995
                    ADDON     =    $D998
                    SYNFND    =    $D9AA
                    OUTDO     =    $DB5C
                    FRMNUM    =    $DD67
                    CHKCOM    =    $DEBE
                    PTRGET    =    $DFE3
                    GETSPA    =    $E452
                    MOVINS    =    $E5D4
                    MOVSTR    =    $E5E2
                    GETADR    =    $E752
                    QINT      =    $EBF2

                    SYNCHK    MAC
                              JSR  SYNFND
                              INY
                              JSR  ADDON
                              <<<

                              ORG  $2E0

02E0: A9 4C         INIT      LDA  #$4C
02E2: 8D F5 03                STA  AMVEC
02E5: A9 F0                   LDA  #<SETSYN
02E7: 8D F6 03                STA  AMVEC+1
02EA: A9 02                   LDA  #>SETSYN
```

```
02EC: 8D F7 03   51              STA  AMVEC+2
02EF: 60         52              RTS
                 53
                 54  ********************************
                 55  *                              *
                 56  *    RWTS2VAR: Bill Parker     *
                 57  *                              *
                 58  *  This section simplifies the *
                 59  * manipulation of the DOS RWTS *
                 60  * buffer at $B4BB.B5BA (46267- *
                 61  * 46522) by splitting it up    *
                 62  * and storing it in two string *
                 63  * variables of user's choice:  *
                 64  *     & RWTS2VAR ) B1$,B2$     *
                 65  *                              *
                 66  ********************************
                 67
02F0: A2 CF      68  SETSYN   LDX  #207
02F2: 86 0D      69          STX  $D
02F4: 86 0E      70          STX  $E
02F6: C9 52      71  CK4RWTS  CMP  #'R'          ;Got RWTS2VAR
02F8: D0 48      72          BNE  NOTRWTS       ;No
                 73          ))   SYNCHK
02FA: 20 AA D9   73          JSR  SYNFND
02FD: C8         73          INY
02FE: 20 98 D9   73          JSR  ADDON
                 73          (((
0301: 20 E3 DF   74          JSR  PTRGET        Create/find descriptor
0304: A9 80      75          LDA  #$80          First part (byte) of buffer
                 76  * GETSPA gets space in himem for string.
                 77  * It also moves the FRESPC pointer down.
0306: 20 52 E4   78          JSR  GETSPA
                 79  * The following builds the descriptor to the
                 80  * string.  It has nothing to do with MOVSTR.
0309: A0 00      81          LDY  #0            Clear index
030B: 91 83      82          STA  (VARPNT),Y Put str len in descrptr
030D: C8         83          INY
030E: A5 71      84          LDA  FRESPC        Get lo addr of strng
0310: 91 83      85          STA  (VARPNT),Y and put in descrptor
0312: C8         86          INY
0313: A5 72      87          LDA  FRESPC+1      Get hi addr of strng
0315: 91 83      88          STA  (VARPNT),Y and put in descrptor
                 89
0317: A0 B4      90          LDY  #)BUF1        Beg of bufr (hi)
0319: A2 BB      91          LDX  #BUF1         Beg of bufr (lo)
031B: A9 80      92          LDA  #$80          First part (byte) of buffer
031D: 20 E2 E5   93          JSR  MOVSTR        Copy bufr to strng storage
0320: 20 BE DE   94          JSR  CHKCOM        Check for comma bet. vars
0323: 20 E3 DF   95          JSR  PTRGET
0326: A9 80      96          LDA  #$80          Prepare 2nd part of buffer
0328: 20 52 E4   97          JSR  GETSPA
032B: A0 00      98          LDY  #0
032D: 91 83      99          STA  (VARPNT),Y
032F: C8         100         INY
0330: A5 71      101         LDA  FRESPC
0332: 91 83      102         STA  (VARPNT),Y
0334: C8         103         INY
0335: A5 72      104         LDA  FRESPC+1
0337: 91 83      105         STA  (VARPNT),Y
0339: A0 B5      106         LDY  #)BUF2        Beg of 2nd part (hi)
033B: A2 3B      107         LDX  #BUF2         Beg of 2nd part (lo)
033D: A9 80      108         LDA  #$80          Len of 2nd part
                 109
033F: 4C E2 E5   110         JMP  MOVSTR
                 111 NOTRWTS  =    *
                 112 *
                 113 ********************************
                 114 *                              *
                 115 * Kill Kontrol Kharcters : VJG *
                 116 *                              *
                 117 * Note: & Command syntax       *
                 118 * recognizes only the first    *
                 119 * character and the ")"        *
                 120 * separator as significant.    *
                 121 * e.g., & R ) B1$,B2$ or       *
                 122 * & V B1$, B2$                 *
                 123 *                              *
                 124 ********************************
```

```
                        125  *
0342: C9 43             126          CMP   #'C'
0344: D0 1D             127          BNE   NOTC
                        128          LDX   #$0          ;Initialize index
0346: A2 00             129  CLOOP   LDA   BUF1,X       ;Load a character
0348: BD BB B4          130          STA   BUF1,X
034B: 9D BB B4          131          CMP   #$0D
034E: C9 0D             132          BEQ   FIXDCTL      ;Protect carriage returns
0350: F0 04             133          CMP   #$20         ;Is it a control?
0352: C9 20             134          BLT   CTLREPL      ;Yes, go fix it
0354: 90 06             135  FIXDCTL INX                ;Increment index
0356: E8                136          BNE   CLOOP        ;End of search?
0357: D0 EF             137          JMP   DATA         ;Yes, go home
0359: 4C 95 D9          138  *
                        139  CTLREPL LDA   #$20         ;Turn it in for a new one
035C: A9 20             140          STA   BUF1,X       ;and put it in the buffer
035E: 9D BB B4          141          BNE   FIXDCTL      ;Always taken
0361: D0 F3             142  *
                        143  *
0363: C9 4C             144  NOTC    CMP   #'L'
0365: D0 19             145          BNE   NOTL
0367: A2 00             146          LDX   #$0          ;Initialize index
0369: BD BB B4          147  JLOOP   LDA   BUF1,X       ;Load a character
036C: 9D BB B4          148          STA   BUF1,X
036F: C9 0A             149          CMP   #$0A         ;Is it a control J?
0371: F0 06             150          BEQ   JREPL        ;Yes, go fix it
0373: E8                151  FIXDJ   INX                ;Increment index
0374: D0 F3             152          BNE   JLOOP        ;End of search?
0376: 4C 95 D9          153          JMP   DATA         ;Yes, go home
                        154  *
0379: A9 20             155  JREPL   LDA   #$20         ;Turn it in for a new one
037B: 9D BB B4          156          STA   BUF1,X       ;and put it in the buffer
037E: D0 F3             157          BNE   FIXDJ        ;Always taken
                        158  *
                        ************************************
                        160  *                          *
                        161  * Here's where Dave Sparks *
                        162  * gets into the act and takes *
                        163  * the data we've placed in the *
                        164  * buffer and cleverly writes *
                        165  * it to disk...            *
                        166  *                          *
                        ************************************
                        168  *
0380: C9 4D             169  NOTL    CMP   #'M'         ;Move command?
0382: D0 18             170          BNE   NOTM         ;Nope, branch ahead
                        171          )))   SYNCHK       ;Move TXTPTR
0384: 20 AA D9          171          JSR   SYNFND
0387: C8                171          INY
0388: 20 98 D9          171          JSR   ADDON
                        171          (((
038B: 20 67 DD          172          JSR   FRMNUM       ;Evaluate starting address
038E: 20 52 E7          173          JSR   GETADR       ;Make it integer in LINNUM
0391: A0 00             174          LDY   #0
0393: B9 BB B4          175  MOVLOOP LDA   BUF1,Y      ;Get byte from buffer
0396: 91 50             176          STA   (LINNUM),Y   ;And save it in file image
0398: C8                177          INY
0399: D0 F8             178          BNE   MOVLOOP
039B: 60                179          RTS
039C: C9 4F             180  NOTM    CMP   #'O'         ;Output command?
039E: D0 2E             181          BNE   STUMPED      ;All out of guesses, goto break
                        182          )))   SYNCHK       ;Move TXTPTR
03A0: 20 AA D9          182          JSR   SYNFND
03A3: C8                182          INY
03A4: 20 98 D9          182          JSR   ADDON
                        182          (((
03A7: 20 67 DD          183          JSR   FRMNUM       ;Evaluate starting address
03AA: 20 52 E7          184          JSR   GETADR       ;Make integer in LINNUM
03AD: 20 BE DE          185          JSR   CHKCOM       ;Move TXTPTR
03B0: 20 67 DD          186          JSR   FRMNUM       ;Evaluate ending address
03B3: 20 F2 EB          187          JSR   QINT         ;Make it integer in FAC
03B6: B1 50             188  SENDCHR LDA   (BYTPTR),Y
03B8: 20 5C DB          189          JSR   OUTDO        ;Send char to output device
03BB: E6 50             190          INC   BYTPTR
03BD: D0 02             191          BNE   BUMPED
03BF: E6 51             192          INC   BYTPTR+1     ;BYTPTR = LINNUM
03C1: A5 A1             193  BUMPED  LDA   LSTBYTLO     ;An address inside FAC
03C3: C5 50             194          CMP   BYTPTR
03C5: A5 A0             195          LDA   LSTBYTHI
```

CP/MUFFIN

```
03C7: E5 51      196             SBC  BYTPTR+1
03C9: B0 EB      197             BCS  SENDCHR
03CB: 4C FB DA   198             JMP  $DAFB      ;Send a C/R and exit
03CE: 00         199  STUMPED    BRK

--End assembly--

239 bytes

Errors: 0
```

```
]LIST

10  REM

  [CP/M]UFFIN BY VAL J GOLDING

WITH DAVE SPARKS AND BILL PARKER
  CALL -A.P.P.L.E. : MARCH 1983

 CONVERTS CP/M TEXT FILES TO DOS

100  GOTO 9000: REM

500  REM

GET TRACK NBR & PATTERN FOR BLOCK

510 TRACK% =  INT (BLOCK%(I,K + 1
    ) / MOD) + 3
520 PTN =  INT ( ABS (BLOCK%(I,K +
    1)) -  INT ( ABS (BLOCK%(I,K
     + 1) / MOD)) *  ABS (MOD) +
    .05) *  SGN (BLOCK%(I,K + 1)
     / MOD)
530  RETURN : REM

1000  REM

WRITE ONE FILE

1010  VTAB 7: PRINT "WRITING ";NU
    MOFBLOCKS * 4;" SECTORS..."
1020  POKE 34,11: HOME
1030  PRINT  CHR$ (4);"MONO"
1040  PRINT  CHR$ (4);"OPENT.";FI
    LNAME$(I)
1050  PRINT  CHR$ (4);"WRITET.";F
    ILNAME$(I)
1060  & OUTPUT ) BFPTR(1),BFPTR(2
    ) - 1
1070  PRINT  CHR$ (4);"NOMONO"
1080  PRINT  CHR$ (4);"CLOSE"
1090  POKE 34,5: HOME : RETURN
1100  END : REM

2000  REM

CALL RWTS TO READ CPM SECTOR

2010  POKE TRACK,TRACK%: POKE SEC
    TR,SECTR%: POKE DRIVE,DRIVE%

2020  VTAB 4: HTAB 1: PRINT "READ
    ING BLOCK:  ";: INVERSE : PRINT
    BLOCK%(I,K + 1);"  ";: NORMAL
     : PRINT " OF:  ";: INVERSE :
     PRINT FILNAME$(I): NORMAL
2030  PRINT "TRACK:          ";:
     INVERSE : PRINT TR%;" ";: NORMAL
     : PRINT "  SECTOR: ";: INVERSE
     : PRINT SECTR%;" ": NORMAL

2040  POKE RWCODE,1: POKE PREGIST
    ER,0
2050  CALL USERRWTS
2060  POKE RWCODE,2: POKE PREGIST
    ER,0
2070  IF CTRLCHR THEN  & CTRLKILL

2080  IF JCTRL THEN  & LINEFEEDKI
    LL
2090  & R ) BUFFR$(1),BUFFR$(2)
2100  RETURN : REM

3000  REM

READ AND WRITE A BLOCK

3010  FOR I = 1 TO LSTFILE
3020 RNUM = 0: IF SNGL THEN I = S
    NGL: REM

3200  FOR J = 0 TO 15: IF BLOCK%(
    I,J + 1) = 0 THEN NUMOFBLOCK
    S = J:J = 15: NEXT J: GOTO 3
    220
3210  NEXT J:NUMOFBLOCKS = 16
3220  IF  NOT  LEN (FILNAME$(I)) THEN
    3710: REM

3300 BFPTR(2) = BFPTR(1)
3310  HOME
3320  VTAB 10: PRINT "DATA LOCATE
    D FROM ";BFPTR(1);" TO ";: REM


3400  FOR K = 0 TO NUMOFBLOCKS -
    1
3410  GOSUB 500: REM

3500  FOR J = 1 TO 4
3510 SECTR% = STBL(PTN,J)
3520  GOSUB 2000
3530  & MOVE ) BFPTR(2):BFPTR(2) =
    BFPTR(2) + 256: VTAB 10: HTAB
    28: PRINT BFPTR(2) - 1
3540  NEXT J: REM

3600  NEXT K: REM

3700  GOSUB 1000
3710  IF SNGL THEN  POKE 34,3: HOME
     : GOTO 6100
3720  NEXT I
3730  END : REM

4000  REM

READ CPM DIRECTORY

4010 TRACK% = 3:SECTR% = 0:ENTNBR
    = 0
4020  FOR I = 1 TO 6: GOSUB 2000:
     REM
```

```
4100 NDX = 1: FOR J = 1 TO 4:ENTN
     BR = ENTNBR + 1
4110 DIR$(ENTNBR) =  MID$ (BUFFR$
     (1),NDX,32)
4120 NDX = NDX + 32:: NEXT J: REM


4200 NDX = 1: FOR J = 1 TO 4:ENTN
     BR = ENTNBR + 1
4210 DIR$(ENTNBR) =  MID$ (BUFFR$
     (2),NDX,32)
4220 NDX = NDX + 32: NEXT J: REM


4300 SECTR% = SECTR% + 6
4310  IF SECTR% ) 15 THEN SECTR% =
     SECTR% - 15
4330  NEXT I: REM

5000  REM

INTREPRET CPM DIRECTORY

5010 K = 0: FOR I = 1 TO 48
5020  IF  MID$ (DIR$(I),2,1) = E5
     $ THEN I = 48: NEXT I: GOTO
     5420: REM
5030  IF  LEFT$ (DIR$(I),1) = E5$
      THEN 5410: REM
CHECK IF DELETED FILE

5040 K = K + 1:FILNAME$(K) =  MID$
     (DIR$(I),2,11)
5050 TEST$ = ( MID$ (DIR$(I),13,1
     )): IF TEST$ (  )  CHR$ (0) THEN
     FILNAME$(K) = FILNAME$(K) +
     "." +  CHR$ ( ASC (TEST$) +
     48): REM
CHECK FOR FILENAME SUFFIX

5060  FOR J = 1 TO  LEN (FILNAME$
     (K)): IF  MID$ (FILNAME$(K),
     J,1) (  ) " " THEN X$ = X$ +
      MID$ (FILNAME$(K),J,1): REM
      REMOVE SPACES

5070  NEXT :FILNAME$(K) = X$:X$ =
     ""
5080 NREC%(K) =  ASC ( MID$ (DIR$
     (I),16,1)): REM  CHECK FOR
      DUMMY FILE

5090  IF  NOT NR%(K) THEN FILNAME
     $(K) = "": GOTO 5410: REM

5200  FOR J = 1 TO 16
5210 BLOCK%(K,J) =  ASC ( MID$ (D
     IR$(I),J + 16,1))
5220  IF BLOCK%(K,J) = 0 THEN J =
     16: NEXT J: GOTO 5410: REM

5400  NEXT J
5410  NEXT I
5420 LSTFILE = K: REM  FALL THRU
     TO 6000

6000  REM

COPY OPTIONS MENU

6010  POKE 35,24: HOME : VTAB 9: PRINT
     "COPY ENTIRE DISK (Y/N) ":: GET
     CHOICE$: PRINT CHOICE$:
6020  IF CHOICE$ = "Y" THEN  GOSUB
     6400: GOTO 3000: REM

6100  HOME : FOR DIR = 1 TO LSTFI
     LE
6110  IF  NOT  LEN (FILNAME$(DIR)
     ) THEN 6130
6120  PRINT " [";DIR;"] ";FILNAME
     $(DIR)
6130  NEXT DIR: REM

6200  PRINT : INPUT "WHICH FILE N
     BR ? [PRESS RETURN TO QUIT]"
     ;CHOICE$:SNGL =  VAL (CHOICE
     $): IF  NOT SNGL THEN  IF  LEN
     (CHOICE$) THEN 6200
6210  IF  NOT SNGL THEN  POKE 34,
     0: END
6220  GOSUB 6400: GOTO 3020: REM


6400  PRINT : PRINT "REMOVE ALL C
     ONTROL CHARACTERS FROM TEXT
     (Y/N) ? ":: GET CHOICE$: PRINT
     CHOICE$
6410  IF CHOICE$ = "Y" THEN CTRLC
     HR = 1: RETURN : REM

6500  PRINT : PRINT "REMOVE CONTR
     OL J'S FROM TEXT (Y/N) ? ";:
      GET CHOICE$: PRINT CHOICE$
6510  IF CHOICE$ = "Y" THEN JCTRL
     CHR = 1
6520  RETURN : REM

9000  REM

INITIALIZATION

9010  HOME : VTAB 12: HTAB 11: FLASH
     : PRINT " LOADING HEX DATA "
     : NORMAL
9020 HX$ = "2E0:A9 4C 8D F5 03 A9
      F0 8D F6 03 A9 02 8D F7 03
     60 A2 CF 86 0D 86 0E C9 52 D
     0 48 20 AA D9 C8 20 98 D9 20
      E3 DF A9 80 20 52 E4 A0 00
     91 83 C8 A5 71": GOSUB 9900
9030 HX$ = "310:91 83 C8 A5 72 91
      83 A0 B4 A2 BB A9 80 20 E2
     E5 20 BE DE 20 E3 DF A9 80 2
     0 52 E4 A0 00 91 83 C8 A5 71
      91 83 C8 A5 72 91 83 A0 B5
     A2 3B A9 80 4C": GOSUB 9900
9040 HX$ = "340:E2 E5 C9 43 D0 1D
      A2 00 BD BB B4 9D BB B4 C9
     0D F0 04 C9 20 90 06 E8 D0 E
     F 4C 95 D9 A9 20 9D BB B4 D0
      F3 C9 4C D0 19 A2 00 BD BB
     B4 9D BB B4 C9": GOSUB 9900
9050 HX$ = "370:0A F0 06 E8 D0 F3
      4C 95 D9 A9 20 9D BB B4 D0
     F3 C9 4D D0 18 20 AA D9 C8 2
     0 98 D9 20 67 DD 20 52 E7 A0
      00 B9 BB B4 91 50 C8 D0 F8
     60 C9 4F D0 2E": GOSUB 9900
9060 HX$ = "3A0:20 AA D9 C8 20 98
      D9 20 67 DD 20 52 E7 20 BE
     DE 20 67 0D 20 F2 EB B1 50 2
     0 5C DB E6 50 D0 02 E6 51 A5
      A1 C5 50 A5 A0 E5 51 B0 EB
     4C F8 DA 00": GOSUB 9900
9070  CALL 736: REM

9080  REM

RELOCATABLE MACHINE CODE
MODIFIED BY DAVID SPARKS

9200 BFPTR(1) = 22016: HIMEM: 220
     16
9210 RWCODE = 45121:SLOT = 46583:
     DRIVE = 46584:
```

Laing Electronics

10382 STANFORD AVENUE, SUITE E-203
GARDEN GROVE, CALIFORNIA 92640
PHONE (714) 534-4216

WIZARD 16 K CARD

By Wisper Micro - 2 Yr. Warranty

$64.00 Plus Shipping

★ HARDWARE FOR THE APPLE II ★

Bold Type indicates a PRICE REDUCTION ! !	LIST	YOUR COST
AMDEK VIDEO 300 MONITOR	249.00	143.00
NEC 12'' HI-RES GREEN MONITOR	285.00	153.00
HAYES MICROMODEM II	379.00	263.00
HAYES SMARTMODEM (RS232)	279.00	199.00
NOVATION APPLE CAT II	389.00	**268.00**
MICRO-SCI APPLE COMPATIBLE DISK DRIVE W/O CONTROLLER	395.00	**259.00**
MICROSOFT Z-80 SOFTCARD CP/M	395.00	**229.00**
MICROSOFT 16K RAM CARD	195.00	69.00
NEC PC-8023A PRINTER	695.00	468.00
OKIDATA MICROLINE 83A PRINTER	995.00	688.00
VIDEX VIDEOTERM CARD - 80 COL.	345.00	228.00
WIZARD 80 COLUMN CARD	249.00	**149.00**

★ DISKS BY THE CASE (100 TO A CASE) ★

BASF 5¼'' SOFT SECTORED	400.00	237.00
DYSAN 5¼'' SOFT SECTORED	600.00	363.00
MAXELL 5¼'' SOFT SECTORED	650.00	303.00
VERBATIM DATALIFE 5¼'' SOFT SECTORED WITH HUB RING	450.00	253.00

★ SOFTWARE FOR THE APPLE II ★

WORD STAR (MICRO PRO CP/M)	495.00	299.00
SUPER SORT (MICRO PRO CP/M)	250.00	152.00
MAIL MERGER (MICRO PRO CP/M)	250.00	152.00
DATA STAR (MICRO PRO CP/M)	295.00	179.00
SPELL STAR (MICRO PRO CP/M)	250.00	152.00
CALC STAR (MICRO PRO CP/M)	200.00	99.00
FORTRAN-80 (MICROSOFT CP/M)	195.00	125.00
BASIC COMPILER (MICROSOFT CP/M)	395.00	252.00
A.L.D.S. (MICROSOFT CP/M)	125.00	80.00

★ PERSONAL SOFTWARE ★

VISIFILES	250.00	179.00
VISIPLOT	200.00	143.00
VISITREND/VISIPLOT	300.00	215.00
VISICALC 3.3	250.00	179.00

★ SOFTWARE PUBLISHING ★

PERSONAL FILING SYSTEM	125.00	76.00
PERSONAL REPORT SYSTEM	95.00	58.00

★ STONE WARE ★

D B MASTER	229.00	147.00
D B UTILITY PACK	99.00	64.00

ACCEPTED PAYMENT - PERSONAL CHECK or MONEY ORDER. NO CREDIT CARDS.
If Personal Check, Allow 10 BANKING DAYS for CLEARANCE.
Prices and specifications subject to change without notice.
Unless otherwise noted, ALL PRICES are PLUS SHIPPING.
SHIPPING & HANDLING — ADD $5.00 UPS SURFACE (DISK DRIVES $7.50 — MONITORS $10.00 — PRINTERS $15.00).
California residents add 6% Sales Tax.

Circle 6 on Reader Service Card

CP/MUFFIN

```
9220 TRACK = 45975:SECTR = 45976:
     MOD = 4:E5$ =  CHR$ (229)
9230 USERRW = 45111:PREGISTER = 7
     2
9240  DIM DIR$(49),FILNAME$(49),N
     R%(49),BLOCK%(49,17): REM

9400  TEXT : HOME : VTAB 2: PRINT
     "[CP/MUFFIN]  CP/M->DOS FILE
      CONVERTER": PRINT : POKE 34
     ,3: INPUT "SOURCE SLOT, DRIV
     E: ";SLOT%,DRIVE%
9410  POKE SLOT,SLOT% * 16: POKE
     DRIVE,DRIVE%
9420  POKE 35,16: VTAB 20: HTAB 1
     1: PRINT : INVERSE : PRINT "
      READING CP/M DIRECTORY ":::
      NORMAL : REM
9600  REM

SET UP 4 SECTOR READ SEQUENCE
PATTERNS

9610 STBL(0,1) = 0:STBL(0,2) = 6:
     STBL(0,3) = 12:STBL(0,4) = 3
9620 STBL(1,1) = 9:STBL(1,2) = 15
     :STBL(1,3) = 14:STBL(1,4) =
     5
9630 STBL(2,1) = 11:STBL(2,2) = 2
     :STBL(2,3) = 8:STBL(2,4) = 7

9640 STBL(3,1) = 13:STBL(3,2) = 4
     :STBL(3,3) = 10:STBL(3,4) =
     1
9650  GOTO 4000: REM
9900  REM

POKE RWTS STRING MACHINE CODE
INTO MEMORY WITH S H LAM ROUTINE

9910 HX$ = HX$ + " N D9C6G": FOR
     I = 1 TO  LEN (HX$): POKE 51
     1 + I, ASC ( MID$ (HX$,I,1))
      + 128: NEXT : POKE 72,0: CALL
      - 144: RETURN
```


Now Available to A.P.P.L.E. Members
For Only

$11.00

The Elementary Apple

by Bill Sanders

This 10 chapter, easy to read illustrated tutorial delves deep into Apple hardware from **power on.**

- A **must** for anyone getting into Apple computers for the first time.
- Teaches you how to do more with your Apple and get the most out of your programs.
- And Much, Much More.

A.P.P.L.E. Orders
21246 68th Ave. S.
Kent, WA 98032
(206) 872-2245

Foreign members add $6.00 for shipping/handling
Washington state residents add 6.8% sales tax
Canadian members, postage being revised

RH ELECTRONICS, INC.

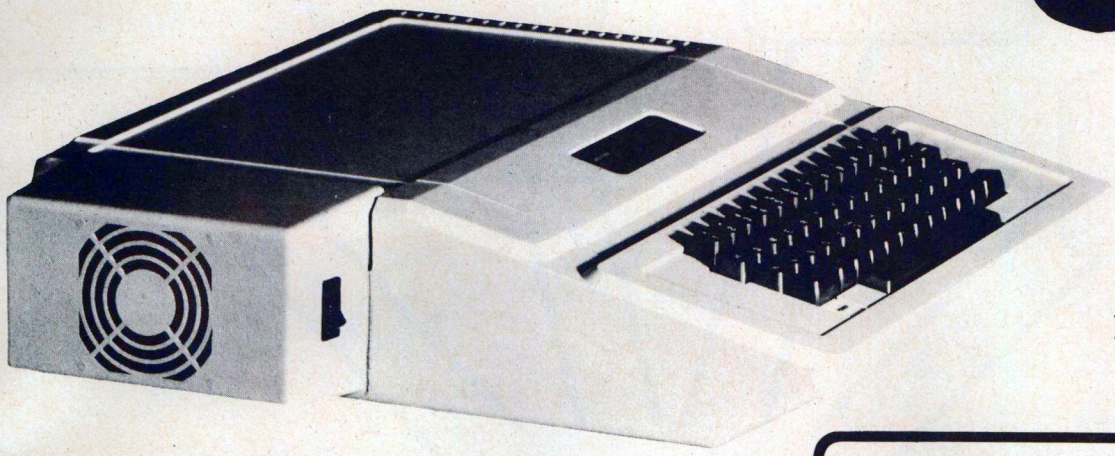
COMPUTER PRODUCTS
DESIGNING • MANUFACTURING
ELECTRONIC ENGINEERING

COPYRIGHT © 1981 — PATENTS PENDING

566 Irelan, Buellton, CA 93427
(805) 688-2047
8:00 TO 5:00 CALIFORNIA TIME

SUPER FAN II™

FOR YOUR APPLE II* COMPUTER

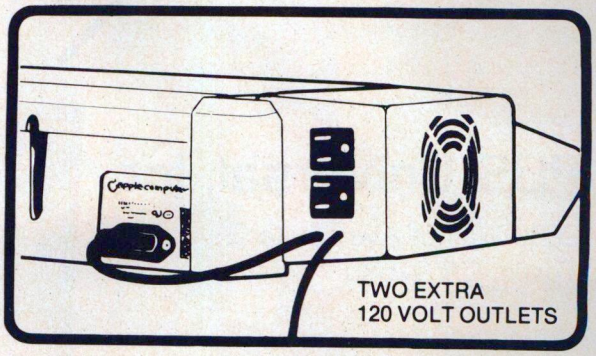


One Year Warranty

$74.95

With Zener Ray™ Protection $109.00

MASTERCARD — VISA



"COOL IT"

- ALSO FITS ON APPLE'S* NEW MONITOR STAND
- RED PILOT LIGHT ON/OFF SYSTEM SWITCH
- CLIPS ON — NO HOLES OR SCREWS • REPLACEABLE SWITCH
- AVAILABLE IN 120V or 240V AND 50/60 HZ • DURABLE MOTOR
- REDUCES HEAT CAUSED BY EXTRA PLUG-IN CARDS
- SOLD WORLD WIDE • UNIQUE 1 YEAR WARRANTY
- TAN OR BLACK COLOR • QUIETEST FAN ON THE MARKET
- INCREASED RELIABILITY — SAVES DOWN TIME AND REPAIR CHARGES
- LOW NOISE DUE TO DRAWING EFFECT OF AIR THROUGH YOUR COMPUTER AND SPECIAL FAN AND MOTOR DESIGN
- TWO EXTRA 120V OUTLETS FOR MONITOR AND ACCESSORIES TURN ON WHEN YOU TURN ON YOUR FAN (NOT AVAILABLE ON 240V MODEL)

SUPER FAN II™ WITH ZENER RAY OPTION $109.00

ZENER RAY™ TRANSIENT VOLTAGE SUPPRESSOR

OUR BUILT IN ADVANCED DESIGN UNIT GIVES
DRAMATIC COST SAVINGS — STOPS ANNOYING DOWN TIME
INSURANCE FROM VOLTAGE SPIKES - GLITCHES

DANGEROUS VOLTAGE SPIKES CAN JEOPARDIZE YOUR COMPUTER SYSTEMS
PROTECT COMPUTER - DISK DRIVE - PRINTER AND MONITOR

NO CUTTING WIRES • WON'T VOID WARRANTY, JUST PLUG IN SUPERFAN II WITH ZENER RAY

OTHER PRODUCTS BY RH ELECTRONICS, INC.

SUPER RAM II™ 16K RAM CARD FOR YOUR APPLE II. 2 YEAR WARRANTY $125

GUARDIAN ANGEL™ AN UNINTERRUPTABLE POWER SOURCE $595

12 VOLT TRANSVERTER 12 VOLT — RUNS YOUR APPLE II COMPUTER AND AND 5¼" DRIVE FROM YOUR CIGARETTE LIGHTER $149

*Registered trademarks of Apple Computer Inc.

DEALER INQUIRIES INVITED

Circle 7 on Reader Service Card

All Logical Paths Lead to

PDS DATADRIVE™ FOR YOUR APPLE II*

great price

15% greater storage

100% **Apple II*** compatible

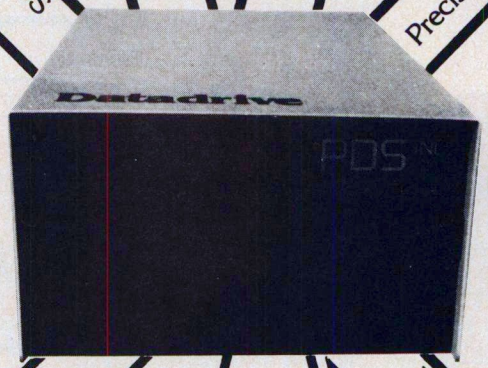
300% track-to-track speed

Outstanding value

Enhancer diskette and 40-track mode

Half-tracking, DOS PASCAL, CPM

Precision lead screen



PDS Universal, Inc.
2630 Walnut Ave., Suite G
Tustin, CA 92680

(714) 730-6772
Telex 18-3511
answer back CSMA

Disconnect Interface Board

Highest standards 8500 hours MTBF

Shugart** mechanism and read/write electronics

When Datadrive is not driving it uses no power

easy interconnect to other computers

super-silent operation long life

state-of-the-arts electro-mechanics

conserve power

*Apple II is a trademark of Apple Computers Inc.
**Shugart is a trademark of Shugart Corporation

Circle 8 on Reader Service Card

JEANNE'S PROCEDURES:

A BEGINNERS GUIDE TO THE PASCAL EDITOR

Terry Reilly

I like Apple's implementation of UCSD Pascal for composing text and writing programs. Wanting to share this capability, I said to my wife: "It's a lot easier and much faster to type that letter on the Apple". You can review it, change it and rearrange it to your liking before it goes to paper. And if not right, it's easier to go back and make changes than it is to retype a new copy from start. "OK, I'll try it," she said, "How does it work?" After a quick and flashy demonstration that stirred enough interest, she gave it a try. I gave her the Pascal manual for reference — bad idea. After several "How's that again" questions, I was directed to write a step-by-step procedure. It helped; I use it myself from time to time, and I hope others will find these procedures useful.

IMPORTANT: Never open the disk drive door while the red "in use" light is on. Interrupting a write operation could damage the contents of your disk.

Start Up

1. Put your disk in Drive 1, turn on the Monitor and Apple. There will be beeps, and disk running, then the greetings will appear on the screen. Note the "current date." If not correct then you should change it.
2. Change date and check on the work file
   [Press] F for FILER
   [Press] D for DATE and enter current date
   [Press] RETURN to keep date
   [Press] E and : to get list of files on your disk
   [Press] RETURN to display files
   (Note: if SYSTEM.WRK.TEXT is listed then this is the file that will come up when you are in the EDITOR mode. If you want to work on a new file (letter) then press N to delete the existing workfile, otherwise do nothing. Of course you can save it under a different name by pressing S for SAVE and then enter a name to save the file under. If you do this, then press N for NEW to clear the work file.)
   [Press] Q to QUIT and FILER and return to the COMMAND line.

Ready to Write a Letter?

1. Enter the EDITOR
   [Press] E for EDITOR
   (Note: if a work file is present this file will be written to the screen. If one is not, you can press RETURN to start a new work file, or type in an existing file name to display it for changes or additions.)
2. Typing Environment
   (Note: this data is saved with each file created and need not be checked each time the EDITOR is called. If this is a new file then do the following)
   [Press] S to SET and
   [Press] E for ENVIRONMENT. Change the data if necessary to:

Auto Indent	FALSE
Left Margin	10
Para Margin	5
Filling	TRUE
Right Margin	70
Token def	FALSE

   [Press] RETURN to save the data
3. Ready to start typing?
   [Press] I for INSERT and type your letter. To make corrections and changes see paragraph 5. When finished
   [Press] Ctrl-C to get out of Insert
   [Press] Q to QUIT and leave the EDITOR
   [Press] U to UPDATE the work file, note that the Command line is now back at the top of the screen.
4. Ready to print your letter?
   Check that the printer paper is aligned and turn the Printer on
   [Press] X to EXECUTE your print output program
   [Press] RETURN for the work file and
   [Press] RETURN for Printer:
   If the Printer "ON LINE" light is on the printer will print the file. If not, then press this button. When printing has stopped; press "ON LINE" to turn the light off, then FF (form feed) to advance the paper. When the paper stops, tear off your copy and check your work. If no errors occur then you are done, go to Cleanup and Turn Off. For additional copies repeat this step.
5. Need to make corrections?
   If you are not in the EDITOR then
   [Press] E for the EDITOR and your letter will appear on the screen. Find and make your corrections.
   [Press] X to EXCHANGE characters.

A.P.P.L.E. Members only

Be the first on your block!

with

Kids & The Apple

Kids & The Apple is an 8½ x 11 full color tutorial textbook written with **clear and easy instructions** at the 10-14 year old age level.

Kids & The Apple will teach kids to program board games, word games, action games, how to store & recall useful data, write drill programs for math, spelling, history and illustrate them with Apple's graphic capabilities. All this in 33 carefully sequenced lessons with explanations, examples, exercises and review questions.

$16.00

A.P.P.L.E. ORDERS
21246 68th Ave.
Kent, WA 98032
(206) 872-2245

Washington State Residents add 6.8% sales tax
Foreign shipping/handling charge: $6.00
Canadian postage being revised

TRANSFER ///
Version 1.1

A valuable Apple / / / utility

Moves sequential text files either way between an Apple][disk (DOS 3.3) and your Apple ///:

VisiCalc data files
Word-processor files
BASIC programs
as Text files
Laboratory-data files

All actions required are performed easily and automatically. You simply select options from menus.
Runs with 1 or 2 disk drives.

Apple /// diskette and manual

$60.00

Mass. residents add $3.00 5% sales tax
Overseas orders add $5.00 shipping

Mind Systems Corporation
P.O. Box 506
Northampton, MA 01061
(413)586-6463

Apple][and Apple / / / are Trademarks of Apple Computer, Inc.
VisiCalc is a Trademark of Personal Software, Inc.

PASCAL EDITOR

[Press] I to INSERT additional characters.
[Press] D to DELETE characters.
[Press] A to ADJUST line position.
[Press] Ctrl-C to QUIT the X I D A modes.
[Press] J to JUMP to B (beginning) or E (end).
[Press] P to PAGE through the letter a screen at a time.
[Press] Ctrl-O to move the cursor up one line, or
[Press] Ctrl-L to move the cursor down one line
[Press] Q to QUIT the EDITOR when all corrections are made, and of course U to UPDATE the work file. Then go to paragraph 4 to print out a new copy.

Cleanup and Turn Off

1. Cleanup is deciding what to do with your letter. You can:
   —leave it on disk as a work file so you can go back to it at a later time, or
   —save it under a different name because you like it, or
   —delete if from the disk. If you save them all, the disk will become full and you will need another disk.
   a. Leave the work file on disk. Go to Turn Off.
   b. Save the work file under a different name?
      [Press] F for FILER
      [Press] S for SAVE, enter the desired name
      [Press] RETURN
      [Press] N for NEW to clear the work file
      [Press] Q to QUIT the FILER and go to Turn Off.
   c. Delete the work file?
      [Press] F for FILER
      [Press] N for NEW to clear the work file
      [Press] Q to QUIT the FILER
2. Turn Off
   Remove your disk and put it in the jacket and in a safe place. Turn off the Monitor, Apple and Printer, and put the dust covers on.

StarLogic Announces Savings on Apple II Compatible Drives

Includes drive, cable, cabinet and standard warranty which includes 90 days parts and labor

5¼" standard disk drive	$205.00
ThinLine half-height disk drive	$185.00
Dual ThinLine drives	$335.00

(Also compatible with Franklin ACE)

TELEPHONE ORDERS ONLY
MASTERCARD, VISA, CASHIER'S CHECK COD ORDERS ACCEPTED
(213) 883-0587

StarLogic

Apple is a registered trademark of Apple Computer, Inc.
ThinLine is a registered trademark of Tandon Corporation

INTRODUCING THE ONLY SOURCE FOR SOFTWARE YOU'LL EVER NEED.

The Software Source is a mail order system which offers the largest and most complete selection of software in the world. Not only do we carry more programs, we also carry more manufacturers. So regardless of the hardware you have, you will not find a more complete selection of mail order software...anywhere.

The Software Source is more than just another "list" of software programs. Once you have decided on the particular programs you want, you order safely and easily from us. No need to contact each vendor directly. In other words... one stop shopping.

Most catalogs only provide a minimal descriptive listing to give you a "hint" of what you're buying. The Software Source gives a complete description of each program to allow you to make the best selection before you order. And we carry all programs instead of a few from "selected" vendors. For example: A leading mail order catalog lists only 9 programs for Reading and Spelling, from 3 vendors. The Software Source carries 47 programs for Reading and Spelling from 13 vendors for most types of hardware.

The Software Source uses a 3-ring, loose-leaf, hard-cover binder that allows you to easily up-date your catalog. That's right, up-date! You will receive one full year's FREE up-dates when you purchase The Software Source. Subsequent up-dates are available at a low subscription rate. You always see the latest program information as it's released, so your catalog is always up-to-date.

The Software Source is mail order so you save...a lot! Not only are our prices lower, but you'll benefit from special sale prices and promotions. More importantly, each time you purchase through The Software Source, you will receive "bonus" points good toward FREE software. Your software dollars go farther.

Toll free numbers, major credit card acceptance and easy to use order forms make The Software Source easy and safe to use.

THE software SOURCE
INCORPORATED
2701 CW 15th St. • Suite 109 • Plano, Texas 75075

Send me The Software Source and one year's updates.

☐ Check enclosed, $26.95. (Includes Postage and Handling).
☐ Bill me
☐ Visa ☐ MasterCard

Card No. _____

Expiration Date _____

_____
Signature of Card Holder

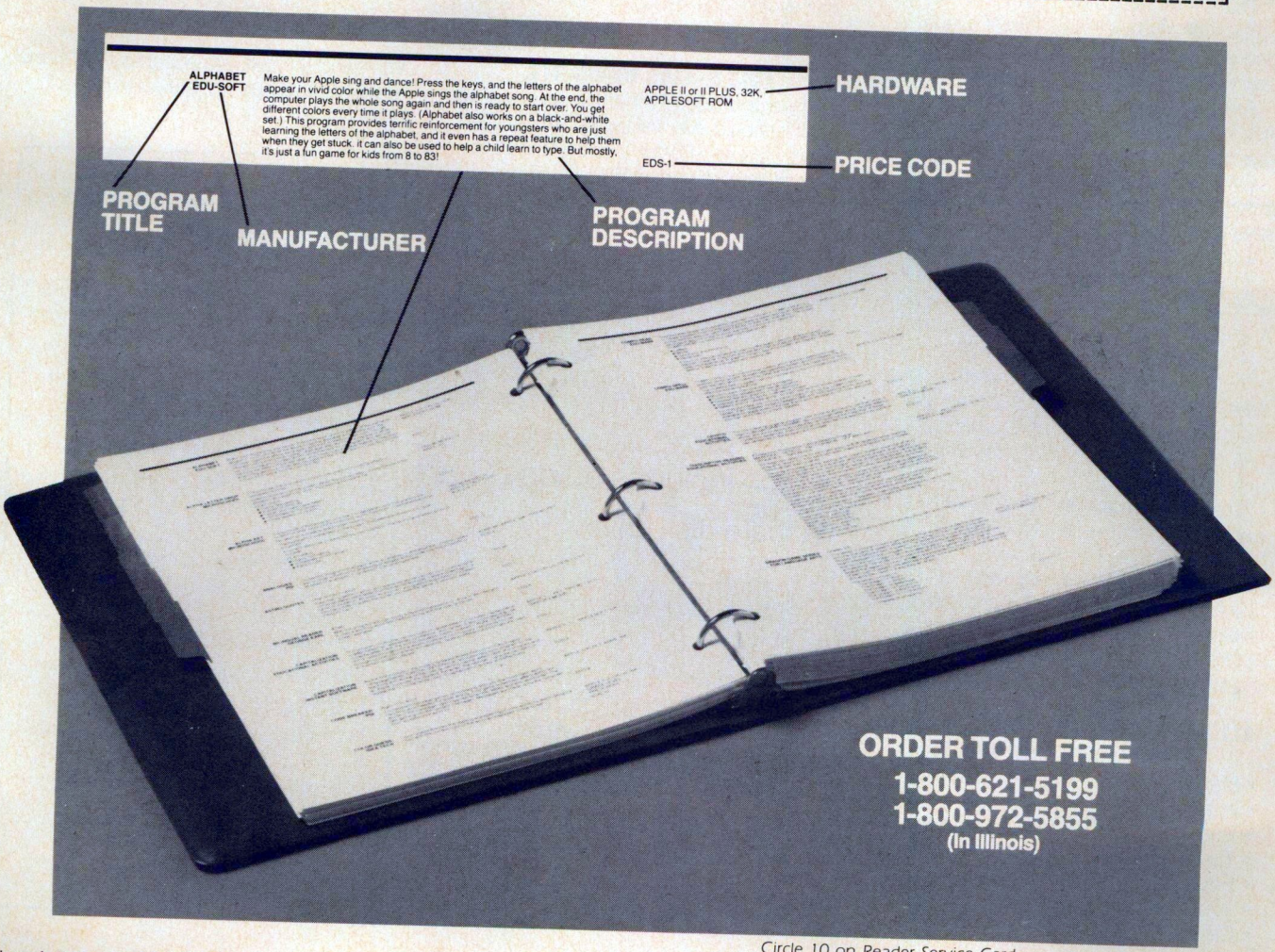
_____
Name

_____
Address

_____
City

_____
State Zip

The Software Source • 2701 CW 15th St. Suite 109 • Plano, Texas 75075



Circle 10 on Reader Service Card

POWERTEXT. THE ONLY FULLY AUTOMATIC WORD PROCESSING SYSTEM FOR YOUR APPLE II.

Now even hunt-and-peck typists produce perfectly formatted letters... memos... presentations... scripts... automatically.

NOW... A COMPLETE SELECTION OF HIGH-PERFORMANCE FEATURES IN TODAY'S MOST ADVANCED WORD PROCESSING TOOL.

The professionals who use PowerText tell us it's changed their whole concept of word processing.

Because this is the system that goes far beyond the write/edit/print/file capabilities of other word processors.

PowerText produces documents that look exactly as you want them to look... *automatically*. From everyday business letters to the most complicated presentations and reports you can devise.

You define the basic formats. PowerText stores them in its style files.

So when you write, you just bat out the words and never worry about where they go on the printed page.

That's PowerText's job.

You just tell PowerText, "I want a letter," and you get a letter...and envelope. With every element just where you want it to be.

Ask for a memo and you get a memo. Ask for a dramatic script and you get a dramatic script...

OK, that's one big PowerText *plus*: the automatic formatting capability that can save you untold time and effort.

But there's still more. A full selection of high-performance features to make your word processing faster, easier, more *automatic* than any system you've ever seen...

Total control editing.

Automatic indents and numbers for outlines, with system-assigned Roman and Arabic numerals, and alphanumerics.

A disk-based editor: file sizes limited only by disk space.

The system automatically maintains a backup copy of the file being edited.

User-definable function keys.

Up to 7 levels of "nested" editing to let you suspend editing one file and move to another.

"Paint mode" editing allows vertical typing for diagrams.

ALL THIS IS AUTOMATIC IN EVERY KIND OF DOCUMENT YOU WRITE.

- Margins
- Indents
- Spacing
- Title page
- Envelope, label
- Justification
- Centering
- Variable pitch
- Pagination
- Table of contents
- Headers and footers
- Footnote Numbers
- Intelligent page breaks

PLUS MANY OTHER VALUABLE FEATURES.

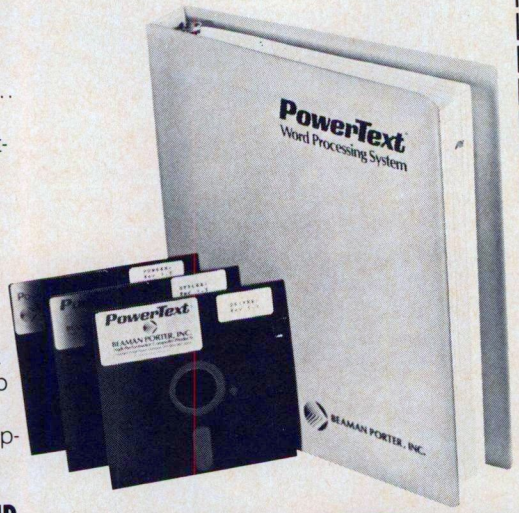
- Built-in form letter capability
- Boxed copy
- Print macros
- Up to 14 columns
- Boilerplate inclusion
- Optional word-by-word cursor move
- Vert. and horiz. border lines
- No limit to document length
- 132-character lines
- Superscripts, subscripts
- Column and line adjustment capabilities
- And much more.

NEW 10-LESSON TUTORIAL.

Our new, streamlined 10-lesson tutorial makes learning PowerText a snap. You'll master it quickly, as you discover how easy and straightforward it is to use. And your system also includes a complete reference manual.

FULL 5-YEAR WARRANTY.

PowerText does everything we say it will do. If it should ever fail to perform as specified, Beaman Porter, Inc. will fix it at no charge, anytime within 5 years after you've bought it.



PowerText

BEAMAN PORTER, INC.
High Performance Computer Products

Pleasant Ridge Rd. Dept. STA-3
Harrison, NY 10528 (914) 967-3504

SYSTEM REQUIREMENTS.

PowerText is a self-contained, bootable system, that supports both serial and parallel printer interfaces. You'll need your Apple II with 64K, 2 disk drives, and a printer.

COMPLETE APPLE II POWERTEXT SYSTEM SOFTWARE PLUS TUTORIAL AND MANUAL $299.

SPECIAL! PowerText *plus* easily-installed PowerCase chip for Apple II upper/lowercase and shift key capability, just $329.95.)

(PowerText for Apple II and III users who have Apple Pascal, $199.)

(PowerText is also available for IBM PC. Write for information.)

Tutorial/manual alone for any version: $25.

Available from selected dealers. Or order directly from Beaman Porter.

VISA and MasterCard accepted. NY State residents add appropriate sales tax.

BEAMAN PORTER, INC., DEPT. STA-3
Pleasant Ridge Road
Harrison, NY 10528

☐ Send _____ complete PowerText system(s) for Apple II @$299. ☐ With PowerCase upper/lowercase chip $329.95.
☐ I have Apple Pascal. Send _____ complete PowerText system(s) for ☐ Apple II or ☐ III @$199.
☐ Send tutorial/manual only, for _____ version @$25.
(NY State residents add appropriate sales tax)
My check or money order for $_____ enclosed.
☐ Or bill $_____ to my:
☐ MasterCard ☐ Visa
(MasterCard only list 4 digits above your name_____)
Card No._____ Exp. Date_____
Name_____
Street _____
City, state, ZIP_____

Circle 11 on Reader Service Card

CUSTOMIZING GPLE ESCAPE TABLES

Edward Nutter

RIGHT at the start I'd like to say that if you own an Apple and don't have the Global Program Line Editor by Neil Konzen, grab your checkbook, an envelope, and a stamp and order it from the folks at *Call -A.P.P.L.E.* Other than a good text editor for developing the broad outlines of a BASIC program, I know of no better program development tool.

Besides such things as insert, delete, and the global search and replace goodies, Mr. Konzen spent the time and effort to include an "escape" function feature. This allows any key to be programmed to execute an immediate mode function, rather like an EXEC file. First one hits the "esc" key, then the key assigned to the desired function and it's done! There are 1152 bytes set aside in GPLE for these escape functions. Since they can call each other like subroutines as well as DOS and ROM machine language routines, this adds up to a very powerful extension to the Apples capabilities.

Included with this system is an escape table full of useful functions such as CATALOG any slot or drive and print the number of free sectors, hide and unhide Applesoft or Integer programs, or print the starting address and length of the last BLOADed program in hex. The functions provided are themselves useful, and provide examples on how to define your own escape functions.

As I played with and enjoyed my new editing capability I began to notice that some of the functions got very heavy use while others were seldom touched. That would be no problem except that I wanted to add some features of my own. GPLE allows escape table functions to be modified in memory. There is also a utility included that modifies GPLE on disk so that it starts up with your changes ready to go, and another utility that prints the current escape table to screen or printer so you can keep notes on some of those nifty "stock" routines.

I wish I had a nickel for every time I typed GOUSB for GOSUB or RETRUN for RETURN. This irritation led to my decision to redefine my keyboard. Esc R became RETURN and esc G became GOSUB. Almost immediately I ran out of room in my 1152 byte buffer. I erased the functions that I had little or no use for, but was soon running out of room again. And what if I wanted to use them later? Redefining a bunch of keys each time I want to exchange functions is just the sort of thing that I don't want to have to do manually. I wanted to be able to change my escape tables "on the fly" so that I could have just the functions I wanted when I wanted them.

Impatience is the midwife of invention, so I started working on a program that would save, load, and even erase escape function tables. Many cups of coffee later I had a program ready that would do exactly that with all three versions of GPLE.

For those of you who, like me, like to jump ahead and type in the program before reading the details, here is how to use it once you've got it in. The first thing up is a menu that gives you the option to save, load or erase an escape table, see instructions, or quit the program. If the load or save options are selected you'll be prompted for first a file name, whether to use the current drive, then slot and drive selections. At that point the program saves or loads the escape table as instructed. The save can take a few seconds to get the table moved before the disk drive starts, so be patient.

The erase option blanks the screen for a few moments so expect that also. The instruction option will print a summary of these instructions. The [ESC] to quit option exits the program, but before it finishes it prints a couple of escape functions that you may add to your escape tables.

The quit option may also be selected at the end of the instructions. These functions, one for PLE.LC and PLE.DM and the other for PLE.48, will replace your current escape table with a BLOADed one. If you include one of these table move functions, a selection of escape table names, and a BLOAD in your escape table, you'll be able to load up new set of functions in seconds with two keystrokes!

Some explanation of the memory usage of GPLE is in order. There are three versions of GPLE. One, PLE.48, locates itself between DOS and its buffers, moving HIMEM down by 4K. For systems with a language card or equivalent RAM card there are PLE.LC and PLE.DM.

PLE.LC is used with the alternate BASIC on the RAM card, and PLE.DM is used when you have DOS on the card. (Cornelis Bongers' program to move DOS there is included with the package). 16K RAM cards for the Apple have 8K of RAM mapped into the space from 57344 ($E000) to 65535 ($FFFF). The remaining 8K is split betweeen two 4K banks, one of which can appear to be at 53248 ($D000) to 57343 ($DFFF) at any one time. PLE.LC and PLE.DM live on bank 1, despite what it looks like on the memory maps on page 32 of the documentation.

Saving an escape table from PLE.48 would be no problem. Its location in memory makes it "visible" to the 6502 simultaneously with BASIC and DOS. Typing "BSAVE filename,A$4B7F,L$480" would tuck it away on a disk. PLE.LC and PLE.DM with their location on the RAM card pose a problem though. PLE.LC occupies part of the same memory space as both BASICs. If you use your motherboard's BASIC to turn on the card by typing "X = PEEK(-16253) the BASIC will disappear as far as the 6502 is concerned, and the computer will "hang."

With PLE.LC there is still a way to save a table from the keyboard. I could have transferred my ROM based monitor and Applesoft to the card using the monitor's MOVE routine. To do that I would have typed CALL-151 RETURN, then C081 N C081 N D000<D000.FFFFM N C088 RETURN.

The two successive references to $C081 write-enable the RAM card and select bank 2 to be in the $D000 to $DFFF space. The next instruc-

—●→

GPLE ESCAPE TABLES

tion copies the ROM BASIC and monitor onto the card. C088 write-protects the card and selects the RAM card's bank 1 to be "visible" to the 6502. At this point Applesoft can't be used because about half of its code (on bank 2) is invisible, but the monitor and DOS can. Now I could BSAVE my escape table from the monitor by typing "BSAVE filename,A$DB7F,L$480" Typing C080 N 3D0G would get me back into the RAM cards BASIC to work on my program. It would be easy to define the whole procedure into escape table functions.

PLE.DM presents a real problem though. The same DOS that must be used to save the escape table occupies the same memory space as the table and Applesoft. Since I couldn't save the PLE.DM escape table from its working location, I decided to move the table down onto the motherboard RAM and then save and load it from there. Now the question is, how to accomplish this?

Among the useful books on the Apple are *Apple II Monitors Peeled* by Apple Computer, and *What's Where in the Apple?* by William F. Luebbert, published by MICRO INK, Inc. Both books contain the location of the monitor MOVE routine ($FE2C). If you're of equal fluency in English and Assembly languages or far more patient than I you can also pluck the information out of the *Apple II Reference Manual.* I called -151 and took a look at MOVE. I found:

```
FE2C - B1 3C    LDA ($3C),Y
FE2E - 91 42    STA ($42),Y
FE30 - 20 B4 FC JSR $FCB4
FE33 - 90 F7    BCC $FE2C
FE35 - 60       RTS
```

The first two instructions move a byte using the indirect indexed method of addressing. LDA means LoaD Accumulator, the accumulator being the main work register of the 6502. The "($3C)" is the location of the first 8 bits of a 16 bit address ($3D is the second 8 bits) that is a pointer to where the byte to be loaded is actually located. It would be, that is, except for the ",Y".

That is how the difference between an indirect address and an indexed indirect address is shown on an assembly listing. "Y" refers to the 6502's Y register. This register contains a value which is added to the address at $3C,$3D to get the effective address that the byte will be fetched from. The STA (STore Accumulator at) instruction works in exactly the same way.

I then looked at FCB4 and found:

```
FCB4    - E6 42   INC $42
    ;adj dest lo byte
FCB6    - D0 02   BNE FCBA
    ;adj hi byte?
FCB8    - E6 43   INC $43
    ;yes
FCBA    - A5 3C   LDA $3C
    ;no. Find out if
FCBC    - C5 3E   CMP $3E
    ;source hi byte needs
FCBE    - A5 3D   LDA $3D
    ;to be adjusted.
FCC0    - E5 3F   SBC $3F
FCC2    - E6 3C   INC $3C
    ;adj source lo byte
FCC4    - D0 02   BNE $FCC8
    ;adj hi byte?
FCC6    - E6 3D   INC $3D
    ;yes
FCC8    - 60      RTS
```

Without going into a lot of details, this subroutine adjusts the pointer

```
]LIST

5  REM
ESCAPE TABLE SAVE BY ED NUTTER
CALL -A.P.P.L.E. : MARCH 1983

10  TEXT : HOME : DEF  FN AD(X) =
     PEEK (X) +  PEEK (X + 1) *
    256:Y =  FN AD(1017): ON (Y <
    ) 36352 AND Y <  ) 46771 AND
    Y <  ) 48896) GOTO 1500:D$ =
     CHR$ (13) +  CHR$ (4):CL =
     - 868:CS =  - 958:BS$ =  CHR$
    (8):KS =  - 16368:KB =  - 16
    384:B1 = 1:B7 = 128:P2 = 511
    :HM =  FN AD(115)
15  A$ = "6:68 68 60 N 6G": GOSUB
    1000: IF HM ) 34304 THEN A$ =
    "304<FCB4.FCC7M N 300:B1 3C
    91 42 N 318:90 E6 60 N 6G": GOSUB
    1000
20  HOME : PRINT  TAB( 10)"ESCAPE
     TABLE SAVER": PRINT  TAB( 9
    )"(C) 1982 - ED NUTTER": PRINT
    : PRINT
40  PRINT "PRESS ";: INVERSE : PRINT
    "?";: NORMAL : PRINT " FOR I
    NSTRUCTIONS": HTAB 8: INVERSE
    : PRINT "ESC";: NORMAL : PRINT
    " TO EXIT
50  VTAB 10: INVERSE : PRINT "S";
    : NORMAL : PRINT "AVE, ";:::
     INVERSE : PRINT "L";: NORMAL
    : PRINT "OAD, OR ";: INVERSE
    : PRINT "E";: NORMAL : PRINT
    "RASE AN ESCAPE TABLE? S";BS
    $;: GET A$: IF A$ =  CHR$ (1
    3) THEN A$ = "S"
60  PRINT A$: ON A$ = "S" GOSUB 1
    00: ON A$ = "L" GOSUB 200: ON
    A$ = "E" GOSUB 600: ON A$ =
     CHR$ (27) GOTO 570: ON A$ =
    "?" GOSUB 500: HOME : GOTO 2
    0
70  VTAB 24: PRINT  CHR$ (7): TAB(
    7)"*** INVALID FILE NAME ***
    ";: FOR T = 0 TO 500: NEXT :
     POP : POP : RETURN
80  INPUT "";ET$: ON ET$ = "" GOSUB
    70: IF (( ASC ( LEFT$ (ET$,1
    )) ) 47 AND  ASC ( LEFT$ (ET
    $,1)) < 60) OR  LEN (ET$) >
    30) THEN  GOSUB 70
90  RETURN
100  VTAB 12: CALL CL: PRINT "SAV
    E AS ";: GOSUB 80: GOSUB 350
150  IF HM < 34305 THEN A$ = "4B7
    F<987F.9CFFM N 6G": GOSUB 10
    00: GOSUB 180: RETURN
170  A$ = "31B:A9 4B 85 43 A9 64 8
    5 42 A9 DB 85 3D A9 64 85 3C
     A9 DF 85 3F A9 FF 85 3E 2C
    88 C0 2C 88 C0 20 00 03 2C 8
    2 C0 60 N 6G": GOSUB 1000: CALL
    795
180  ONERR  GOTO 400
190  PRINT D$"BSAVE";ET$;DP$;",A$
    4B7F,L$480": POKE 216,0: RETURN
200  VTAB 12: CALL CL: PRINT "WHI
    CH TABLE ";: GOSUB 80: GOSUB
    350: ONERR  GOTO 400
240  PRINT D$"BLOAD";ET$;DP$: POKE
    216,0
250  IF HM < 34305 THEN A$ = "987
    F<4B7F.4FFFM N 6G": GOSUB 10
    00: RETURN
```

values at $42,$43 and $3C to $3F that are used by the program that calls it. It also tells the caller when to quit by setting a flag known as the "carry" flag to one when the pointers to the "start" address are equal to the "end" address. Since I couldn't use it where it was I decided to use MOVE to move itself down to where I could get at it when the RAM card was turned on. This is only necessary for PLE.LC and PLE.DM. First the pointer adjust routine less the RTS instruction is copied to $304. Next a modification of the routine at $FE2C is wrapped around it so that it becomes a single purpose MOVE. This happens in line 15.

At this point I should explain an unusual technique I used in this program. The "Lam" method of generating machine language routines from within a BASIC program is well known. Briefly, a string is defined that is the same as what one would type if they were using the monitor. This string is POKEd into the keyboard buffer area, where keystrokes are stored before RETURN is hit (See line 1000).

At location -144 in the Apple monitor lives the routine that executes any monitor input lines in the keyboard buffer. There is a problem in that it is not designed to return control to a running BASIC program. Most of the running return calls designed to overcome that do not allow CALL -144 to be used from within a subroutine. This can limit program flexibility. On page 77 of Apple][Monitors Peeled there is an explanation of why this is so. To quote it:

"Calling" the Monitor Command Processor implies that return will take place to the calling program. However, the driver part of the Monitor Command Processor is not designed to operate in that fashion, so a short machine language program is required to allow exit back to the calling program. . . . In the sample, the three byte machine language routine is placed in page two (at $2FC), but it may be placed anywhere desired.

The routine (in assembler notation) is PLA, PLA, RTS. The PLA's each pop one return address off the "stack" on page 1 ($100,$1FF). The RTS (ReTurn from Subroutine) returns control to BASIC. At the beginning of line 15 is a short routine that initializes and uses that method of exit from monitor to Applesoft. I stick the return routine at 6, 7, and 8 on page zero instead of the top of page 2. I wrote a short program to search the ROM's for that three byte combination, in the hope that I could use it. There were none.

I decided around this time to move the escape tables of all three versions to the same point to simplify the disk access routines and to allow interchange of escape tables between versions. This contains a minor peril I'll explain later. I used the top of Hi-Res page 2, from $4B7F to $4FFF as the "buffer" for the escape tables.

Program Breakdown:

Line 10 is mostly initialization stuff. Using the variable names as mnemonics makes the program more readable (I think) and speeds up such routines as line 1000. GPLE changes the pointer address at 1017,1018 ($3F9, $3FA) used by [Ctrl-Y] in the monitor. I ran the various GPLE's to find what these pointer values were. The variable Y is set equal to whatever the pointer address is — if Y is not equal to any of the expected

—●→

```
260 A$ = "200:2C 89 C0 2C 89 C0 6
    0 N 200G N DB7F<4B7F.4FFFM N
     200:2C 82 C0 60 N 200G N 6G
    ": GOSUB 1000: RETURN
350 VTAB 15: CALL CS: PRINT TAB(
    10)"THIS DISK? (Y/N) ";: GET
    A$: PRINT A$: ON A$ = CHR$
    (27) GOTO 390: IF A$ = "Y" THEN
    DP$ = "": RETURN : REM FROM
    WITHIN THIS ROUTINE PRESS ES
    C FOR THE MENU
360 VTAB 17: CALL CL: PRINT TAB(
    15)"SLOT: ";: GET SL$: PRINT
    SL$: ON SL$ = CHR$ (27) GOTO
    390: IF VAL (SL$) < 4 OR VAL
    (SL$) > 6 THEN 360
370 VTAB 18: CALL CL: PRINT TAB(
    15)"DRIVE: ";: GET DR$: PRINT
    DR$: ON DR$ = CHR$ (27) GOTO
    390: IF VAL (DR$) < 1 OR VAL
    (DR$) > 2 THEN 370
380 DP$ = ",S" + SL$ + ",D" + DR$
    : RETURN
390 A$ = "": POP : RETURN
400 IF PEEK (222) = 4 THEN VTAB
    21: PRINT "DISK WRITE PROTEC
    TED:": PRINT "REMOVE PROTECT
     TAB AND PRESS A KEY": GOSUB
    450: POKE KS,0: WAIT KB,B7:A
     = PEEK (KB) - B7: VTAB 21:
     ON A = 27 GOTO 20: CALL CS:
     RESUME
410 IF PEEK (222) = 6 THEN VTAB
    22: PRINT CHR$ (7)"THAT FIL
    E NOT ON THIS DISK!";: FOR I
     = 0 TO 1000: NEXT : HTAB 1:
     CALL CL: POKE 216,0: GOSUB
    450: GOTO 460
420 IF PEEK (222) = 8 THEN VTAB
    22: PRINT CHR$ (7)"I/O ERRO
    R:": PRINT "          CLOSE D
    RIVE DOOR OR INIT DISK": FOR
    I = 0 TO 1000: NEXT : VTAB 2
    2: CALL CS: POKE 216,0: GOSUB
    450: GOTO 460
430 IF PEEK (222) = 9 THEN VTAB
    22: PRINT "URP!";: FOR I = 0
     TO 25:X = PEEK ( - 16336):
     NEXT I: PRINT "  I'M STUFFE
    D!!";: FOR I = 0 TO 1000: NEXT
    : HTAB 1: CALL CL: POKE 216,
    0: GOSUB 450: GOTO 460
440 VTAB 20: PRINT "DOS ERROR #"
    ; PEEK (222): POKE 216,0: GOSUB
    450: GOTO 460
450 PRINT TAB( 10)"PRESS (ESC)
    TO EXIT": PRINT TAB( 7)"ANY
     OTHER KEY TO CONTINUE": RETURN

460 POKE KS,0: WAIT KB,B7:A = PEEK
    (KB) - 128: POKE KS,0: ON A =
    27 GOTO 570: GOTO 20
500 HOME :A$ = "": PRINT TAB( 1
    6)"USE WITH": PRINT TAB( 7)
    "GLOBAL PROGRAM LINE EDITOR"
    : PRINT TAB( 8)"(C)1982 - B
    Y NEIL KONZEN": PRINT
510 PRINT "TO SAVE YOUR CURRENT
    ESCAPE TABLE PRESS": PRINT "
    'S' OR (RETURN) TO ACCEPT TH
    E DEFAULT.": PRINT : PRINT "
    TO LOAD A PREVIOUSLY SAVED E
    SCAPE TABLE": PRINT "PRESS '
    L'.": PRINT : PRINT "TO ERAS
    E THE CURRENT ESCAPE TABLE "
    : PRINT "PRESS 'E'.": PRINT
```

GPLE ESCAPE TABLES

values, the program jumps to Line 1500 and ends. If we get past that point we assume that GPLE is running, and continue on to where HM is set equal to the value of HIMEM:. This tells the program which version of GPLE is in use.

If PLE.LC or PLE.DM are running, the copy of the monitors MOVE routine will be needed. Line 15 places that routine at $300 (768). Lines 20 through 50 are the main menu. Line 60 sends control to the function you've selected, then loops back to the menu.

The SAVE Function:

Line 100 asks you for a file name to save your escape table under. The subroutine at 80 checks for 1) no name selected, 2) a file name starting with a number, and 3) a file name over 30 characters long. If any of these conditions are true you go to line 70 where you are chided for your error and sent back to the main menu. If your file name is ok you're sent to the subroutine starting at line 350. There you are asked if you want to use the last disk drive accessed. If you press ESC you go back to the main menu via line 390. A "Y" answer returns you to line 100. Any other gets you prompts to select a slot and drive. An ESC response to either of these also sends you back to the menu. Line 380 assembles your responses into a Default Parameter string. At this point you return to the end of line 100. Line 150 finds out if PLE.48 is the version running, and if it is, uses the monitor's version of MOVE to copy the escape table to $4B7F. If PLE.LC or PLE.DM are running, the part of the program that required most of the coffee comes up. When the monitor's command interpreter routine starts looking at a line full of addresses it places them into certain sets of locations on page zero.

The move command in line 150 would be placed like this:

$43,$42 < $3D,$3C . $3F,$3E M
location
4B 7F < 98 7F . 9C FF M value

The move routine at $300 also needs these locations to be initialized before it is called. I tried using the following string: "3C:7F DB FF DF 00 00 7F 40" to set up those locations the same way a 4B7F<DB7F.DFFF command would have. It didn't work. Apparently the monitor uses those locations at $3C to $43 in all its command parsing functions. All I found in those locations later were references to themselves. Therefore, I had to pull out my 6502 book and look up the op-codes to do the set up of those registers from machine language.

In the command string in line 170 the "A9's" are LDA instructions that load the accumulator with the number immediately following. The "85's" are STA's that store the accumulator in the zero page address immediately following. The first part of the command, the 24 bytes up to but not including the 2C, have the same effect as 4B64<DB64.DFFF. Yes, I know that it should be 4B7F DB7F.DFFF. I'll discuss that in a moment. The next six bytes place $C088 on the Apple's address bus twice. This turns on the RAM card with bank 1 on, with the card mapped into $D000,$FFFF. The "20 00 03" calls the MOVE subroutine at $300. When this returns the current escape table has been copied to $4B7F,$4FFF. The next three bytes place $C082 on the address bus. That

```
520  PRINT "YOU MAY USE THE DEFAU
     LT SLOT & DRIVE OR": PRINT "
     SPECIFY NEW ONES.  FOLLOW TH
     E MENU.": PRINT
530  PRINT "FOR EXAMPLES OF ESCAP
     E FUNCTIONS THAT": PRINT "WI
     LL REPLACE THE CURRENT ESCAP
     E TABLE,": PRINT "AND TO EXI
     T THE PROGRAM, PRESS (ESC)."

540  VTAB 23: PRINT "      PRESS A
     NY KEY TO CONTINUE": POKE KS
     ,0: WAIT KB,B7
550 A =  PEEK (KB) - B7: POKE KS,
     0: IF A <  > 27 THEN  RETURN

570  HOME : PRINT "   FOR MOVING
     TABLES TO PLE.LC & PLE.DM": PRINT
     "THE SEQUENCE IS:": PRINT
580  PRINT "'";: INVERSE : PRINT
      CHR$ (93);: NORMAL : PRINT
     ":200:2C 89 C0 2C 89 C0 60 N
     200G N DB7F<4B7F.4FFFM N 200
     :2C 82 C0 60 N200G3D0G";: INVERSE
     : PRINT "M": NORMAL
590  PRINT : PRINT : PRINT "   FO
     R MOVING TABLES TO PLE.48": PRINT
     "THE SEQUENCE IS:": PRINT : PRINT
     "'";: INVERSE : PRINT  CHR$
     (93);: NORMAL : PRINT ":987F
     <4B7F.4FFFM N 3D0G";: INVERSE
     : PRINT "M": NORMAL : PRINT
     : END
```

```
600  VTAB 12: PRINT  TAB( 10)"ARE
      YOU ";: FLASH : PRINT "SURE
     ";: NORMAL : PRINT "? (Y/N)
     ";: GET A$: PRINT A$: IF A$ <
     > "Y" THEN  RETURN
610  HGR2 : ON HM > 34304 GOSUB 2
     60: ON HM < 34305 GOSUB 250:
      TEXT : POKE  - 16300,0: RETURN

1000  FOR I = P2 + 1 TO P2 +  LEN
     (A$): POKE I, ASC ( MID$ (A$
     ,I - P2,B1)) + B7: NEXT : CALL
      - 144: RETURN
1500  VTAB 10: PRINT  TAB( 11): CHR$
     (7);"GPLE NOT INSTALLED": END
```

```
]LIST

80   REM THINK!!! BY DOV FREUND---
      SO WHAT DID YOU EXPECT ?
82   POKE 1013,76: POKE 1014,165: POKE
      1015,214
84   POKE 106,7: LOMEM: 2051:A = 1
     .02529952E + 14
86   LOMEM: 4013: FOR I = 2059 TO
      1E23
88   IF  PEEK (I) = 83 THEN I = 1E
     24: GOTO 92
90   POKE I,8: NEXT
92   POKE 2048 +  PEEK (2049),0: POKE
      2048 +  PEEK (2049) + 1,0
94   END
96   REM SO WHAT DID YOU EXPECT ?
98   REM BY DOV FREUND
100  REM CALL -A.P.P.L.E.
```

causes the RAM card to select bank 2 and return control to the motherboard ROM. "60" is an RTS instruction. Line 1000 gets it into memory and CALL 795 runs it.

I could adapt the program to different hardware and software configurations, but I can't control what you do to your disks, therefore the disk error handling routines that start in line 400. See lines 180 and 200 for the ON ERR GOTO's. The messages in the 400 series lines are pretty self explanatory. I was overcome by whimsy in line 430. Please forgive me. Line 400 will keep looping to 190 unless you press ESC or remove the write-protect tab. The others send you back to the menu. If there are no errors, line 190 saves the escape table on the disk selected.

I promised I'd tell you why I moved the extra 27 bytes in line 170. The truth is I don't know why it's neccessary. I played around with this section for a long time, moving the entire GPLE, including routines that printed the contents and location of everything. As near as I can tell, the escape table does start at $DB7F. If the extra bytes are not included, the front end of the table is truncated, and when loaded back up, GPLE doesn't recognize it. I did come up with a couple of ways around that, but you can work them out for yourselves. To save the entire table, include the extra bytes.

The LOAD Function:

This works very much like the save function. You're prompted for a table name, asked to select a drive, and (if you don't mess up) a table is BLOADed to $4B7F. Line 250 does a straightforward monitor move to PLE.48. The move to PLE.LC and PLE.DM is easier than the move from. The RAM card may be write-enabled while the Motherboard ROM is read enabled. That's why you can so easily move Applesoft and monitor to the card as I described earlier. "2C 89 C0" translates to "BIT $C089" which selects bank 1 and write-enables the card. "DB7F<4B7F. 4FFFM" does the move, and "BIT $C082" write-protects the card and selects bank 2.

The ERASE Function:

This is easy. Line 600 asks if you really want to do this. If you do, HGR2 erases the table buffer area, then the appropriate call to 250 or 260 writes zeros over the escape table. The text 1 screen is turned back on and it's back to the menu.

I now have most of my reserved words for BASIC, Applesoft, and DOS on escape tables. Two functions insert ":PRINT" or ":PRINT:PRINT" in the middle of a literal if I misjudge my formatting. That's one way to keep your longer prompt messages under control. My number keys now load up alternate escape tables, including the default one that restores their catalog/disk space functions. One caution that I referred to earlier, some of the "stock" functions are specific to the GPLE version that they come with. The erase PLE function and disk space functions are for sure. If you try to exchange and use those you may end up rebooting. My special function tables are "relocatable" in that respect. I know you're licking your chops in anticipation, so have fun making your Great Escape!

"THE ULTIMATE TERMINAL PACKAGE"
FOR STANDARD APPLE DOS!

Send **ANY** size or type of file: TEXT (even Random!), BINARY, APPLESOFT and INTEGER with complete error checking. • Extremely enhanced terminal emulation system supporting baud rates up to 9600 baud! • Up to 26 macro libraries — each containing complete operational parameters, along with the usual phone numbers and macro strip!

AFFORDABLY PRICED AT **$129.95**

southwestern data systems™
P.O. BOX 582-C • SANTEE, CA 92071 • **619/562-3221**

Circle 28 on Reader Service Card

DON'T BE AFRAID OF YOUR SOFTWARE!

Give Your Apple The Friendliest Programs Around, From SUPER SOFTWARE SYSTEMS!

WE'VE JUST GIVEN BIRTH TO TRIPLETS!

1) SUPER-SHAPER: The ALL NEW Hi-Res shape-table graphics package that lets you draw anything, right from the keyboard. Combine drawings up to 29,000 points, display multiple images, trace off the screen, rotate and enlarge, and use different colors together. Create composite shape tables which you can use in your OWN programs. Much more. Great for creating character sets, business graphics, art, anything! No hex or other math required. Super-simple to operate.
Special Introductory Price: ONLY $25.00!

2) SUPER-PRINTER: Make your Epson MX series *printer do what YOU want it to. Simple menu commands let you select and mix options to access all of the typefaces and print styles available on these amazing printers! Run Super-Printer BEFORE your word processor, calc or other program, to get the style and size type you want. Compatible with most other programs. No funny printer codes to learn. Just select what you want and GO!

Also includes Addresser: Maintain a list (on disc) of addresses. Print an envelope in seconds. Just select the address, insert the envelope (requires friction feed) and it's done. Automatically prints your return address (from disc). Includes tips on using friction feed the easy way! Unbelievably easy to use.
Special Introductory Price: ONLY $20.00!

3) SUPER-DISC MUNCH: The ultimate disc editor. Examine and rewrite anything on your discs. Easy search, menu driven. ASCII dump with lower case conversion available anytime. Create illegal filenames, modify DOS, change a word in a program. Anything! A great way to learn the ins and outs of DOS.
Special Introductory Price: ONLY $35.00!

All discs include friendly written documentation, in PLAIN ENGLISH.

All programs require: 48K, DOS 3.3 and ROM Applesoft.

*Best when used with Type III, but works with all MX series printers.

All discs run under DOS 3.3, with NO copy-protect gimmicks. All programs listable and copyable. WE DO NOT SELL LOCKED SOFTWARE!

Several "freebie" entertaining programs included with each disc, just for fun.

Questions? Please call us at: (617) 739-2011

(Apple, DOS 3.3 and Applesoft are trademarks of Apple Computer, Inc. Epson is a trademark of Epson America, Inc.)

Yes! I want the friendliest software around!

Please send me:

☐ Super-Shaper $25.00
☐ Super-Printer $20.00
☐ Super-Disc Munch $35.00
Postage+Handling $ 1.50
TOTAL

Make check or M.O. payable to: Super Software Systems

Send to:

Super Software Systems
1334 Commonwealth Ave. Suite 20
Allston, MA 02134

Allow 2 weeks for delivery. Mass. Res. add 5% sales tax.
cam1

Super Software Systems
THE SIMPLE SOLUTION

SIXTH FINGER™

The automatic repeat key for your Apple II*

$9.95

Also introducing our new adjustable unit with on/off feature

$12.95

- Adapts your Apple; all keys held down will repeat after the perfect delay.
- Great for program editing, word processing & VisiCalc.*
- Easy for anyone to install. Plugs onto the Apple's encoder board (all rev. 7 or later and late rev. 6 with encoder board).

(30 day trial)

Regular unit	$9.95
Adjustable unit	12.95
Add shipping	1.00
Overseas shipping	3.00

Cal. res. add 6% tax

To order or request free instruction sheet use order form or call (213) 675-4706

O Visa O Check or money order enclosed
O MasterCard O Free instruction sheet

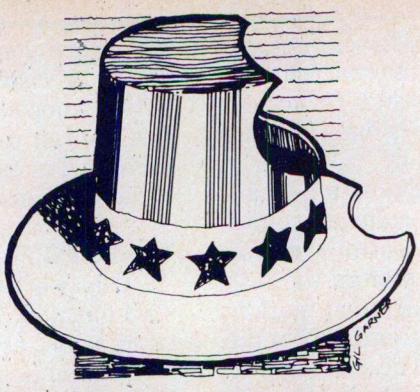
Amount enclosed $ ____ For ____
City ____ St. ____ zip ____
Address ____
Name ____
Sign. ____
Card # ____ Exp. ____

Ivers Specialties
Dept. CA-2
12841 Hawthorne Blvd.
Box 524
Hawthorne, Ca. 90250

Dealer Inquires Invited

*Apple is a registered trademark of Apple Computer Co.
*VisiCalc is a registered trademark of VisiCorp.

Circle 17 on Reader Service Card



BITE INTO YOUR TAXES

with the TAX BREAK optimizer and your Apple*

Forget about researching new tax laws and the confusion and anxiety associated with preparing your 1982 personal income tax return. **TAX BREAK ANNUAL** calculates your lowest tax easily and quickly.

Written by an IRS-Enrolled Agent with over 24 years of experience, **TAX BREAK ANNUAL** automatically examines all loopholes, benefits, deductions, special income averages and limitations. It evaluates over 120 possible reporting options then finds and displays the best way to get the lowest tax on your individual return. Then it shows how your return compares with a typical taxpayer in your income bracket and scores your return for audit potential.

TAX BREAK PLANNER contains all the features of **TAX BREAK ANNUAL.** And then some. Use your **TAX BREAK PLANNER** to design an investment program, then evaluate its impact on your current . . . and future . . . tax status. Do "What if" simulations with real estate, stocks, bonds, annuities, interest rates, etc. . . . then measure their impact on your tax liability.

TAX BREAK ANNUAL is available for $129.95 + $2 shipping & handling (California residents add $7.80 sales tax).

TAX BREAK PLANNER is available for $179.95 + $2 shipping & handling (California residents add $10.80 sales tax). Both programs are tax deductible.

See your local dealer or order direct.

***Apple II, II+, IIe, or III. A Stand alone program. VisiCalc not required. Runs on standard DOS 3.3. Needs only one disk drive and a monitor. Printer not required. Apple is a trademark of Apple Computer, Inc.**

PROFORMA SOFTWARE
2706 Harbor Bl. Suite 200
Costa Mesa, Ca. 92626
(714) 641-3846

Circle 18 on Reader Service Card

Invest 10% & Receive 35% to 150%

That's the kind of speed return the Pascal Speedup System provides. Boosts in graphics and "heavyweight" business software for Apple II/II+ outperforms the competition!

STELLATION TWO

The Lobero Building P.O. Box 2342
Santa Barbara, Ca. 93120
(805) 966-1140 Telex 658439

Circle 15 on Reader Service Card

MAKING IT LAST

Bob Hanson

IN "Pitch and Rhythm on the Apple," *Call -A.P.P.L.E.* June '82, Mr. Bender was looking for a way to produce long musical tones for scores which contained half and whole note values. In other words, how do you make music last? That's easy — make the words first! Seriously, it would be nice to have a routine that could play long notes for slow pieces of music like "White Christmas".

I have managed to devise such a routine, and would like to share it with the *Call -A.P.P.L.E.* readers. But first, a little background:

Examine Listing 1. This is the tone generator that I used. It was supplied by Apple on the *Lemonade Stand* program (a game supplied on DOS 3.2 system masters, but disappeared with the DOS 3.3 masters). Listing 1A contains the decimal values necessary to POKE the routine in from BASIC. To use this routine, you POKE 768 with the duration (time-count), and CALL 770. The routine plays the note and subtracts 1 from location 769 until the value of location 769 equals zero, when the program exits.

With this routine, the longest tone possible (POKE 769,255) is about .8 seconds, which is not an awful long time if you need to generate a tone for any considerable length of time.

In order to obtain notes with time-counts greater than 255, we need two memory locations — one to hold the number of 256-count loops we need (we'll call it HI), and the second will store the remainder of the count, if any (we'll call it LO). As an example, suppose a note requires a time-count of 542. HI would equal INT(542/256), or 2. LO would equal the difference between our time-count and HI=256, or 542-2*256 = 542-512 = 30. As you can see, HI*256+LO will equal your original time-count.

LISTING 1A
POKE VALUES FOR LISTING 1

POKE MEM LOC	VALUE
770	173
771	48
772	192
773	136
774	208
775	5
776	206
777	1
778	3
779	240
780	9
781	202
782	208
783	245
784	174
785	0
786	3
787	76
788	2
789	3
790	96

Technical Corrections to "Making It Last" will be printed in the April Call -A.P.P.L.E.

Listing 1

```
                    ********************************
                    *                              *
                    *    Long Note  Subroutine     *
                    *        by Bob Hanson         *
                    *                              *
                    * Call -A.P.P.L.E. : MARCH 1983 *
                    *                              *
                    * See accompanying article for *
                    *   instructions to POKE PITCH *
                    *   and DURATION values into   *
                    *   memory.                    *
                    *                              *
                    ********************************
                    *

                    *
                    SPEAKR    =     $C030
                    *

                              ORG   $300

                    *
                    TONE      DS    1          ;Pitch of note (DEC 768)
                    TCTHI     DS    1          ;Hi byte of time-count (DEC 769)
                    TCTLO     DS    1          ;Lo byte of same (DEC 770)
                    *
0303: AD 30 C0      SPTOGLE   LDA   SPEAKR     ;"Push" on the Apple's speaker
0306: 88            PLAY      DEY              ;Subtract 1 from Y-reg (counter)
0307: D0 05                   BNE   *+7        ; and branch if non-zero
0309: CE 02 03                DEC   TCTLO      ;Subtract 1 from counter
030C: F0 09                   BEQ   HICHK      ;If zero, go check hi byte
030E: CA                      DEX              ;Subtract 1 from X-reg
030F: D0 F5                   BNE   PLAY       ;Continue playing if non-zero
0311: AE 00 03                LDX   TONE       ;Load X-reg with pitch value
0314: B8                      CLV              ;Clear overflow flag for branch
0315: 50 EC                   BVC   SPTOGLE    ;Start playing again
0317: CE 01 03      HICHK     DEC   TCTHI      ;Subtract 1 from hi byte
031A: D0 EA                   BNE   PLAY       ;Continue playing if non-zero
031C: 60                      RTS              ;RETURN
--End assembly--    29 bytes   Errors: 0
```

MAKING IT LAST

The Routine Itself

My modification routine can be seen in Listing 2, with the corresponding decimal POKE values in Listing 2A. Upon examination, the only differences are a single changed byte and 2 extra instructions at the end.

The program behaves much like Listing 1, with one exception. When LO becomes zero, the two extra instructions are executed. One is subtracted from HI, and the result is examined. If it is non-zero, the routine plays the chosen note for another count of 256. If the result is zero, the routine quits.

To use from BASIC: Let TONE be the type of note we want to play, and TIME will be the time-count for the note.

So:

```
30 HI = INT (TIME/256) : LO =
   TIME -HI * 256: HI = HI + 1:
   POKE 768, TONE : POKE 769,
   HI: POKE 770, LO: CALL 771
```

A word of explanation about HI = HI + 1. With my modification, after 1 is subtracted from HI, HI holds the number of 256-count loops *remaining* to be played. The "+1" serves to act as a kind of flag which tells the routine when to keep playing and when to stop. If TIME=256, then HI = 1, and my routine will work like the routine in Listing 1.

With the addition of the extra memory location, our longest time-count is now 65279. With that as our time-count, our tone will now play continously for two minutes and 45 seconds. You could get a longer tone if you manually POKE 769,0:POKE 770,255, but unless you're timing eggs, I thing 65279 is long enough, don't you?

OK, back to music. Here again is Mr. Bender's note chart, complete for all values of all notes. Please note that his previously published values are a bit smaller. This is to compensate for the small, but noticable, delay that occurs while BASIC calculates HI & LO and POKEs them into memory.

There you have it — music literally at your microchip's fingertips! Only your imagination holds you back! Alright, you future Beethovens, let's see what you've got...

LISTING 2

```
*302L

0302- AD 30 C0  LDA   $C030
0305- 88        DEY
0306- D0 05     BNE   $030D
0308- CE 01 03  DEC   $0301
030B- F0 09     BEQ   $0316
030D- CA        DEX
030E- D0 F5     BNE   $0305
0310- AE 00 03  LDX   $0300
0313- 4C 02 03  JMP   $0302
0316- 60        RTS
0317- 4C EA 03  JMP   $03EA
031A- 60        RTS
031B- C9 80     CMP   #$80
031D- B0 06     BCS   $0325
031F- 29 3F     AND   #$3F
0321- 69 20     ADC   #$20
0323- 49 E0     EOR   #$E0
0325- 48        PHA
0326- 2C 63 C0  BIT   $C063
0329- 10 FB     BPL   $0326
```


A.P.P.L.E. Members Only

The Most Exciting Apple Adventure Game Around

EAMON

Now A.P.P.L.E. offers members these two specials:

ALL ABOUT EAMON

32 Eamon disks plus 1 free disk containing program listing from **Anatomy of an Eamon Adventure** (March, 1983 Call —A.P.P.L.E.)

Regular price $128.00

All About Eamon Special Price $95.00

EAMON SPECIAL

Select any 10 Eamon Adventures (From 32 Total)

Regular Price $40.00

Eamon Special Price $35.00

Both offers expire April 30, 1983

A.P.P.L.E. ORDERS

21246 - 68th Ave. S.

Kent, WA 98032

(206) 872-2245

Washington State Residents Add 6.8% Sales Tax

Foreign Members Add $18.00 shipping/handling charge to **All About Eamon Special**; Add $12.00 to **Eamon Special.**

LISTING 2A

POKE VALUES FOR LISTING 2

POKE

MEM LOC - VALUE

771 - 173
772 - 48
773 - 192
774 - 136
775 - 208
776 - 5
777 - 206
778 - 2
779 - 3
780 - 240
781 - 9
782 - 200
783 - 208
784 - 245
785 - 174
786 - 0
787 - 3
788 - 76
789 - 3
790 - 3
791 - 206
792 - 1
793 - 3
794 - 208
795 - 234
796 - 96

DATA VALUES FOR "SINGIN IN THE RAIN"

PITCH,DURATION	PITCH,DURATION	PITCH,DURATION
172,173	172,173	172,173
86,461	86,436	86,353
102,38	102,38	102,173
115,83	152,173	115,83
128,38	136,436	152,533
152,497	102,38	172,83
172,173	152,173	172,83
128,461	128,436	86,353
128,28	152,38	102,73
115,83	128,173	115,83
102,38	152,533	152,533
86,497	172,83	172,173
172,83	152,38	86,173
152,38	128,436	86,353
128,461	115,38	172,173
115,38	102,173	136,83
102,83	86,533	136,173
86,353	172,83	102,163
102,173	152,38	172,173
86,173	128,436	172,173
86,436	115,38	
102,38	102,173	
115,83	86,533	
152,533	172,173	
172,173	128,436	
86,436	115,38	
102,38	102,173	
115,173	86,353	
152,533	102,173	
172,173	86,173	
86,353	86,436	
102,173	102,38	
115,83	115,173	
152,533	152,533	

DATA VALUES FOR "WHITE CHRISTMAS"

PITCH,DURATION	PITCH,DURATION	PITCH,DURATION
152,665	192,161	152,329
145,161	128,329	152,329
152,161	115,161	152,161
162,161	152,665	115,329
152,161	145,161	128,161
145,665	152,161	96,1001
136,161	172,161	192,161
128,497	192,161	172,161
115,329	172,1330	152,329
102,161	152,665	152,329
96,161	145,161	115,161
86,161	152,161	204,161
96,161	162,161	204,161
102,161	152,161	204,161
115,161	145,665	192,1001
128,1001	136,161	
192,161	128,497	
172,161	115,329	
152,329	102,161	
152,329	96,161	
152,161	102,161	
115,329	115,161	
128,161	128,1001	
192,329	192,161	
192,329	172,161	

𝅗𝅥	A	𝅘𝅥𝅮		𝅗𝅥			𝅗𝅥		𝅗𝅥.	𝅝			
𝅘𝅥𝅮	B			𝅘𝅥𝅯			𝅘𝅥𝅮		𝅘𝅥𝅮.	𝅗𝅥	𝅗𝅥.	𝅗𝅥	𝅗𝅥.
𝅗𝅥	C	𝅘𝅥𝅯	3	𝅘𝅥𝅮	3	𝅘𝅥𝅮.	𝅗𝅥	3	𝅗𝅥.	𝅗𝅥	𝅗𝅥.	𝅝	𝅝.
TEMPO	88	56	77	119	161	182	245	329	371	497	749	1001	1505
	94	53	73	113	153	173	233	313	353	473	713	953	1433
	100	50	69	107	145	164	221	297	335	449	677	905	1361
	106	47	65	101	137	155	209	281	317	425	641	857	1289
	112	44	61	95	129	146	197	265	299	401	605	809	1217
	118	41	57	89	121	137	185	249	281	377	569	761	1145
	124	38	53	83	113	128	173	233	263	353	533	713	1073
	130	35	49	77	105	119	161	217	245	329	497	665	1001
	136	32	45	71	97	110	149	201	227	305	461	617	929
	142	29	41	65	89	101	137	185	209	281	425	569	857
	148	26	37	59	81	92	125	169	191	257	389	521	785
	154	23	33	53	73	83	113	153	173	233	353	473	713
	160	20	29	47	65	74	101	137	155	209	317	425	641
	166	17	25	41	57	65	81	121	130	178	281	377	569

When bites aren't enough,
core gives you more.

"All About Utilities"
Reviews of current utility software

Complete utility program listings also a foldout poster listing all utility data.

CORE, the quarterly that fills its pages with all the information on a single topic. **CORE** carries expert columns and general interest articles in each journal. Subscribe now and also receive the **HARDCORE COMPUTIST** eight times a year. Get in on an open discussion of every facet of the Apple including how to do and undo copy protection, make backups of locked software, use various copy utilities.

HARDCORE is available only through this ad. In all, receive both **CORE** and **HARDCORE** one issue a month for only $20.00.

SUBSCRIPTION
$20 U.S.A, $29 CANADA, $33 MEXICO, $42 ELSEWHERE
SAMPLES U.S.A. $5 ELSEWHERE $8
U.S. FUNDS ONLY
NO CREDIT CARDS

DEPT. N-1
P.O. BOX 44549 A
TACOMA, WA 98444
(206) 581-6038

NAME _____
ADDRESS _____
CITY _____ STATE _____
ZIP _____

Circle 16 on Reader Service Card

MAKING IT LAST

DATA LISTING FOR "JAVA"

PITCH,DURATION	PITCH,DURATION	PITCH,DURATION	PITCH,DURATION
115,53	192,53	121,53	145,53
121,53	172,53	121,53	172,53
115,53	145,53	145,113	192,53
115,53	172,53	172,113	145,113
145,113	192,53	192,53	1,74
172,113	172,113	192,53	86,113
192,53	172,53	172,53	86,113
192,53	192,53	145,53	115,113
172,53	172,113	172,53	115,113
145,42	192,53	192,53	172,113
172,53	172,113	172,113	172,113
192,53	172,53	115,53	230,233
172,113	192,53	121,53	172,53
172,53	172,53	115,53	172,53
192,53	128,53	115,53	172,53
172,113	172,53	145,113	172,53
172,53	192,53	172,113	192,113
192,53	172,113	192,53	192,113
172,113	115,53	192,53	145,317
172,53	121,53	86,53	
192,53	115,53	172,53	
172,53	115,53	86,53	
145,53	145,113	86,53	
172,53	172,113	96,113	
192,53	192,53	172,53	
172,113	192,53	192,53	
115,53	172,53	172,113	
121,53	145,53	172,53	
115,53	172,53	192,53	
115,53	192,53	172,113	
145,113	172,113	172,53	
172,113	115,53	192,53	
192,53	128,53	172,53	

'YOUR'
GLOBAL
Program Line Editor
in
ROM

NOW THAT YOU HAVE THE BEST WHY NOT PUT IT WHERE IT BELONGS. PERMANENTLY IN YOUR APPLE'S MEMORY!

Send us a copy of your configured PLE. LC or PLE. 48 and we will put it on a two ROM set. The GPLE ROM set can then be installed in your M.C. ROMPLUS card and can be ready for your immediate need.

Due to the size of PLE. LC and PLE. 48 the ROM set will not hold the last 222 bytes of the PLE. LC macro table or the last 37 bytes of the PLE. 48 macro table. However, the complete macro table can still be redefined once it is installed in ram memory.

$38.50

PLEASE ADD $3.00 FOR SHIPPING
Visa or MasterCard Accepted

SOFT CTRL SYSTEMS
BOX 599, WEST MILFORD, NJ 07480
201-728-8750

Circle 19 on Reader Service Card

DATA VALUES FOR "ANCHORS AWEIGH"

PITCH,DURATION	PITCH,DURATION
192,418	152,202
512,202	128,202
128,202	115,310
115,310	152,94
152,94	115,418
115,418	96,418
96,418	86,202
86,202	128,202
128,202	96,512
96,512	115,418
115,418	96,202
96,202	115,202
115,202	128,202
128,202	115,202
115,202	102,202
102,202	96,202
96,202	152,94
136,202	1,22
115,202	128,40
86,202	136,94
96,202	128,94
102,202	172,94
128,202	1,22
145,202	128,40
172,202	136,94
192,418	128,94
	192,418

The Executive Secretary

- **Document editor**
- **Powerful printing program**
- **Flexible database**
- **Mail list merge for business letters**
- **Pre-printed forms manager**
- **Electronic mail**
- **Alphabetical indexer for books and theses**

INTEGRATED means that all of the above is included in the $250 suggested retail price, a combination that would cost hundreds of dollars more—IF you could find a set of programs that would work together in the first place.

INTEGRATED means that every part of the package follows the same rules. There's no need to learn one set of keypresses for the editor and a separate set of rules for the database.

THE EDITOR: Auto-configures to 40 or 80 columns on the Apple IIe, supports a wide variety of 80 column boards, or runs in 40 column mode on older Apples. A coupon for a free Revision 7 LCA is included for older Apples.

THE DATABASE: Our users tell us that the Electronic Card File is worth the price of the entire package. It's not really, but with its ease of use, its multiple alphabetic and numeric sorts, its report options that include totals and subtotals, it is superior to many databases costing well over $100.00.

PRE-PRINTED FORMS MANAGER: Use a sequence of special embedded commands to define the row and column positions of the blanks on pre-printed forms. Our template helps you. Then use mail-list merging to combine information from your database with your pre-printed forms.

INTEGRATED

ELECTRONIC MAIL: With a Hayes Micromodem you can transfer your documents to another computer—or to your local typesetter for printing.

SPELLING CHECKER: Easier to use than any other spelling checker. Ours comes with 10,000 words and a capacity for 15,000 more. Or you can start your own list from scratch. Documents are scanned, and you are shown misspellings in context. Correct them on-the-fly, or add words to the dictionary as you scan. A $75.00 option.

MAIL LIST MERGE: You can write documents in the editor that call for information from the Electronic Card File and merge the information to produce customized forms and form letters. Conditional printing allows you to skip words, blocks of text, or entire letters based on criteria that you define in each card in the card file. And you can also draw information from other popular databases, from Visicalc, and from your own custom programs.

RELIABLE means that earlier versions of this software have been in the field for years on the Apple.

THE PRINTER: For more demanding output needs, you can embed format commands in your document while you edit and let our reformatting printer program handle all of your layout concerns. More than 40 embedded commands allow you to control every aspect of layout, including pagination, binding margins, flexible page headers, volume-length documents, outline indentation, and far more. You can employ electronic shorthand so that commonly used phrases are represented by three or fewer keystrokes. And if your printer is capable of it, you can do justification by incremental spacing, boldface, and more.

RELIABLE

RELIABLE means that thousands of users have logged more than 1,000,000 hours of use on the Apple version.

RELIABLE means that we back you up after the sale with a hotline number printed right on the program disk. If your dealer doesn't know the answer, we do.

ALPHABETICAL INDEXER: This unique tool allows you to specify page numbers for each occurrence of a term and to produce a properly formatted alphabetical index of those terms. Two levels of indexing are supported, so that entries like 'beans, navy' and 'beans, green' are handled properly with two levels of indentation.

ORDER BY PHONE

Available at your local dealer or call us at

(612) 929-7104

We accept VISA/MasterCard

Ask for The Executive Secretary at your local dealer.

SOF/SYS, INC.

4306 Upton Ave. South, Minneapolis, MN 55410

Circle 20 on Reader Service Card

FROM THE CREATORS OF THE DATA DRIVE®
COMES APPLETTE 1® AND APPLETTE 2®

SLIMLINE DRIVES

Available At CompuShack Stores

Applette 1® and Applette 2® are 100% Apple II compatible. Halftracking, DOS, PASCAL, and CP/M®. 300% faster track to track speed with 15% greater storage capacity on a 40 track mode with enhancer diskette. TEAC® mechanism and read/write electronics. Direct shaft drive, metal band positioner, photo coupler write-protected sensor. 10,000 lifetime hours, and more. **One year warranty on all parts and labor.**

Headquarters Telex: 18-3511

TAVA CORP.

Answer Back CSMA

(714) 730-6772

*DATA DRIVE, APPLETTE 1, APPLETTE 2, and TRUMP CARD are registered trademarks of TAVA Corporation, respectively

*TEAC is a registered trademark of TEAC Corp

* CP/M is a registered trademark of Digital Research, Inc

Circle 21 on Reader Service Card

A Call -A.P.P.L.E. TECHNIQUE:

CONVERTING NEW TO FP TYPE FILES

W.T. Doyle

A bug in the Applesoft NEW command causes it to initialize the end of program pointer (EOP = $AF, $B0) one byte too high. This introduces a permanent off-set in the EOP and appends one meaningless byte to any program initialized with NEW. These two errors survive additions, deletions, and replacement of lines, and even disk SAVE and LOAD operations. Although they are not normally troublesome (or even visible), the errors become important when one program is to be appended to another. The malady is thoroughly diagnosed by Golding in his article Applesoft from Bottom to Top (*Call -A.P.P.L.E. in Depth No.1,* p. 17), but the remedy prescribed there is not reliable.

The method given assumes that the location (EOP -1) will always contain a non-zero byte for a NEW program. It is easy to see that this is not the case. After initialization with NEW, (EOP -1) always contains the byte that was originally in location $803 when the NEW command was issued. This byte could be anything, including zero, depending on the usage of this area of memory before the NEW command was given. It could even be inadvertently set to zero while entering a program. For example, in the case discussed by Golding, this byte would be zero if line number 100 in his Figure 3 had been chosen to be 256 instead. Location (EOP-1) can not tell us for certain whether a file was initialized with a NEW or an FP command.

Nevertheless, Golding's general method may be used if we transfer the test to location (EOP -4). This location always contains a zero byte for a NEW type file and a non-zero byte for an FP type file. The following single line command performs an unambiguous test of file type and converts NEW files to FP type files.

```
ND = PEEK(175) + 256*PEEK(176)
:IFPEEK(ND-4) = 0 THEN ND =
ND-1:POKE175,ND-256* INT
(ND/256):POKE176,INT(ND/256)
```

Similarly, the following single line command performs a test of initialization and prints out the file type.

```
X$(0) = "FP":X$(1) = "NEW":
PRINTX$(PEEK(PEEK(175) + 256*
PEEK (176)-4 ) = 0)
```

This report can often spare us the trouble of loading, checking, and converting each file, when we have a number of file variants descended from a common parent file. Both the convert and report commands are particularly convenient when implemented as PLE escape commands.

Finally, it is worth noting that we can easily determine the file type and convert NEW to FP files without any special commands at all. The file type can be determined by simply LOADing the file and then DELeting every line. Then if EOP = 2051 it is an FP file, if EOP = 2052 it is a NEW file. If the file is a NEW type file, it may be converted to an FP file by simply renumbering it. Since renumber programs work line-by-line, they automatically reset EOP correctly. If, for some reason, you do not want to renumber the program, you need not actually change any line numbers. Simply pick any line number and renumber it alone, onto itself.

A.P.P.L.E. Members Only

You've Seen the Rest
Now Get the Best
GLOBAL Program Line Editor

Global PLE has all these fine new features to further ease your editing tasks:

- Global edit, search and replacement
- Compatible with 80 column cards
- Typeahead buffer
- 1152 bytes of Escape functions definable from keyboard and nestable
- Page list mode
- Completely transparent to user
- Loads in regular RAM or Language card.

Plus all of the powerful standard PLE commands and features

$38.50

A.P.P.L.E. Orders
21246 68th Ave. S.
Kent, WA 98032
(206) 872-2245

Washington state residents add 6.8% sales tax
Foreign members add $12.00 shipping and handling charge
Canadian members, postage being revised

How to Save Money with the SSM TransModem 1200.™

Cut down your phone time. The TransModem 1200 cuts long distance costs by sending and receiving electronic information faster.

The TransModem 1200 runs at the most widely used baud rates—110, 300, or 1200—so you can easily communicate with any computer, including other personal computers, public information services, and timeshare systems.

You don't even have to set the data transfer speed. The TransModem 1200 automatically adjusts the baud rate to match the speed of any modem on the other end of the line.

You can't find a better equipped modem for the money. The TransModem 1200 completes calls where other modems cannot get through. It dials and answers high speed Touch Tone™ or Pulse dial telephones automatically. It stores telephone numbers of up to 32 digits, enabling you to dial MCI Advantage™ or Sprint® automatically. And Transend™ software is available, so you don't have to do any programming.

Put it to work today. The TransModem 1200 works with any computer, including the IBM PC. Cable and serial interfaces are available for the Apple II/IIe® and Apple III®.

Ask your local SSM dealer for the TransModem 1200. Delivery is immediate so you can start saving money now. Satisfaction is guaranteed or your money back.

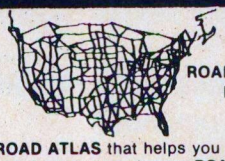
SSM
Microcomputer Products, Inc.

TransModem 1200 and Transend are trademarks of SSM Microcomputer Products, Inc. Touch-Tone is a trademark of AT&T. MCI Advantage is a trademark of MCI. Sprint is a registered trademark of Southern Pacific Communications Co., a Southern Pacific subsidiary. Apple II/IIe and Apple III are registered trademarks of Apple Computer Corporation.

Circle 23 on Reader Service Card

ANNOUNCING
A NEW WAY TO SAVE TIME, MONEY AND GAS

START YOUR NEXT TRIP WITH A COMPUTER LISTING SHOWING THE BEST ROUTE.



ROADSEARCH MAP

- **ROADSEARCH IS A COMPUTERIZED ROAD ATLAS** that helps you plan trips. You get driving routes, mileage, travel times, fuel usage and more. **ROADSEARCH** contains a database of 406 cities/road junctions and 69,000 road miles. A powerful program computes the shortest route between cities in the USA and Canada. Other programs let you develop routes which may be longer, but more suitable to your needs.
- **AIR TRAVELERS** - a unique subroutine computes flying time between cities.
- **EASY TO USE.** Back-up copies are allowed. Requires an Apple II plus with DOS 3.3. **ROADSEARCH** also carries a **15 DAY MONEYBACK GUARANTEE.**
- **STOP USING THE *"OLD FASHIONED"* METHOD** of tracing routes on a map. It is tedious, time consuming, and inaccurate. Use your computer to plan your next route. The printed output is an excellent companion on any trip.

Order **ROADSEARCH** from your dealer or directly from Columbia Software. Disk plus manual is only $34.95. Add $1.50 for shipping and handling. MD residents add 5% state tax.

- - - - - - - - - - MAIL COUPON OR CALL TODAY - - - - - - - - - -

COLUMBIA SOFTWARE (301) 997-3100
P.O. Box 2235 T, Columbia, MD 21045

Yes, I want to use my computer to *SAVE TIME, MONEY AND GAS.* Please send me **ROADSEARCH** for $34.95 plus $1.50 shipping. If I am not satisfied, I may return it for a full refund.

( ) check ( ) Master Charge ( ) VISA ( ) send free info

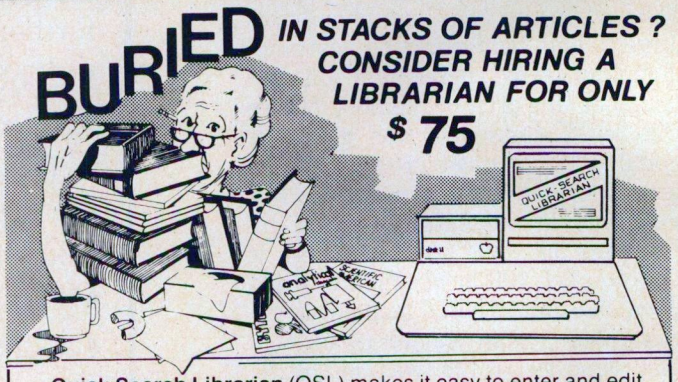
Card Number Expiration Date

Name

Address

City State Zip

Circle 22 on Reader Service Card



Quick Search Librarian (QSL) makes it easy to enter and edit your journal references, search for articles, and print or sort a list of articles using the 48K APPLE* II + computer. Important QSL features include:

- Two keystrokes select any one of 255 keywords or any one of 255 journal titles.
- Four lines available for listing authors, title and/or comments.
- Powerful data base screen editing, copying and merging features.
- Average search speed is 50 articles/second with multiple criteria; average sorting speed is 40 articles/second when sorting on 3 fields.
- Typically, 1000 articles can be stored on a single disk.
- Includes sample data base and tutorial for *Scientific American*, 1981.

VISA or Mastercard orders accepted. QSL manual available separately for $5. (Price of manual deductible later with purchase of QSL software.) Add $1.50 for shipments made in U.S.A.

Circle 24 on Reader Service Card

* Trademark of Apple Computer, Inc.

INTERACTIVE MICROWARE, INC.
P.O. Box 771, Dept. 48
State College, PA 16801, (814) 238-8294

DELETE THAT INCUBUS!

H. S. Bill Balsam

INCUBUS? What's that? "Any person or thing that oppresses or burdens," says definition 2 in *Webster's Collegiate Dictionary* If you're lucky, you've never had extra false lines mysteriously insert themselves into your Applesoft program, garbage that you could *not* delete in the normal way. It is nonsense, a burden, a monkey on your back. I recently had one. I asked a quite knowledgeable professional how to remove it, and he referred me to others as well qualified as he. Before I asked them, I decided to have a go at figuring how to delete that incubus. I finally came up with a very simple method. On the way to it, however, I wasted much time on developing two deletion techniques requiring machine language operations.

Incidentally, how did or does this alien creep in? In my case, I believe it was connected with my lengthening, by modifications, a copied Hi-Res graphics program so that it intruded into Page 1.

There are other ways for garbage to appear. Val Golding suggests a few ways: tabbing to a printer more than 40 columns without first turning off the video display, and POKEing 32,XX and 33,YY such that (XX + YY) is greater than 40. Sometimes you can bomb your whole program so it is un-salvageable.

How do you prevent such garbage from occurring? By being *perfect!* In other words, no way! As Pogo the comic strip opposum declared long ago, "I have met the enemy and he is us!" I suggest that you can fight your errors by posting and following this rule: SAVE BEFORE RUNNING.

Fundamentally, the ML deletion method is to find the offending bytes in memory (using the EXAM routine in *Applesoft In Depth No. 1,* page 9, or an equivalent) and Monitor-MOVE the subsequent part of the program to over-write and replace them. A variant is to adjust the hexadecimal line number equivalents so as to make the Applesoft program bypass the incubus. (See the above mentioned reference, page 15, for a good exposition of BASIC statement structure.) This does not actually remove it, but effectively "buries" it. These Monitor-using methods are effective but require many steps. However, there is a much quicker method, one that others may have previously used and have not bothered to report. That is, use the DELete editing command.

Lines 10 and 0 in Fig. 1 constitute the incubus I encountered. It is the last part of line 1710, the next to last line of the program. I tried to delete it in the usual cursor-tracing way by POKEing 33,33, LISTing 1710, running the cursor through to just beyond the GOSUB 850, and hitting [RETURN]. When I re-LISTed, there is was, ugly as ever! When I simply deleted line 1710 by entering 1710 and [RETURN], planning to re-enter it by running the cursor over it, the incubus attached itself to the preceding line, 1700! Like the grin Lewis Carroll's Cheshire cat, it refused to disappear. Using Neil Konzen's Program Line Editor (PLE), the same nil result.

So here's my quick solution in "How to..." form.

Procedure

1. Display the incubus-plagued line XXX on the screen by LISTing it.
2. Enter immediate-execution command DEL XXX,YYY, where YYY is any number between XXX and that of the next program line.
3. Re-enter line XXX sans incubus in memory in the usual cursor-tracing way.
4. Check by re-LISTing XXX: the incubus is gone! See Fig. 2.

Note: Obviously, such cursor-tracing is facilitated if you have previously POKEd 33,33 or if you have an editor such as PLE installed and have used EDIT to display XXX in 40 columns in step 1.

So the next time you get an incubus, take heart! DELete it!

Figure 1: Before

```
]LIST1710-

1710  IF E = 8 THEN  PRINT "DRIVE
      DOOR IS OPEN!": PRINT "PLEA
     SE CLOSE! AND PRESS A KEY ";
     : GOSUB 530: GET A$: PRINT :
      GOTO 850
10  COLOR=  USR  LET  USR  LET  USR
     NEXT
0
1720  PRINT "ERROR #"E" OCCURRED"
     : END

]DEL1710.1711
```

Figure 2: After

```
]LIST1710-

1710  IF E = 8 THEN  PRINT "DRIVE
      DOOR IS OPEN!": PRINT "PLEA
     SE CLOSE! AND PRESS A KEY ";
     : GOSUB 530: GET A$: PRINT :
      GOTO 850
1720  PRINT "ERROR #"E" OCCURRED"
     : END
```

What's eating your Apple®?

Find out with Apple-Cillin II™

If you use your Apple for your business or profession, you probably rely on it to save you time and money. You can't afford to guess whether it is working properly or not. Now you don't have to guess. Now you can find out with Apple-Cillin II.

Apple-Cillin II is the comprehensive diagnostic system developed by XPS to check the performance of your Apple II computer system. Apple-Cillin II contains 21 menu driven utilities including tests for RAM memory, ROM memory, Language Cards, Memory Cards, DISK system, Drive Speed, Keyboard, Printer, CPU, Peripherals, Tape Ports, Monitors and more. These tests will thoroughly test the operation of your Apple, and either identify a specific problem area or give your system a clean bill of health. You can even log the test results to your printer for a permanent record.

Apple-Cillin II works with any 48K Apple system equipped with one or more disk drives.

To order Apple-Cillin II and to receive information about our other products, call XPS Toll-Free: ***1-800-233-7512.***

In Pennsylvania:

1-717-243-5373

Apple-Cillin II: $49.95. PA residents add 6% State Sales Tax.

XPS

XPS, Inc.

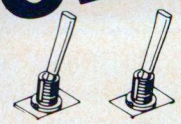
323 York Road
Carlisle, PA 17013

800-233-7512
717-243-5373

Apple is a registered trademark of Apple Computer Inc.

Circle 25 on Reader Service Card

slot swappers



Tired of inconvenience, loose connections and costly burned out chips because software and hardware are slot dependent? The solution has arrived! **Slot Swappers** interchanges any two slots (except 0) at the flip of a switch! Incredibly useful for hard & floppy disk controllers and for multiple printers.

- No wiring, just plugs in
- Installs in minutes
- Easy access to switches
- No chips needed
- No special software required

Johnson Associates
P.O. Box 1870
Phoenix, AZ 85001
(602) 979-4554

Visa/Mastercharge accepted. $2.00 handling charge. AZ residents add 5% tax.

Circle 27 on Reader Service Card

IMAGINATOR™

PROFESSIONAL 3D GRAPHICS PROGRAMS

Imaginator™ is an exciting new series of professional quality 3D graphics programs. Imaginator will let anyone interested in computer graphics create, edit, manipulate and display 3D graphics images. You do not have to be a programmer to use Imaginator. Three dimensional objects can be created without drudgery using Imaginator's powerful graphics editor. Once the objects have been created they can be manipulated and displayed in a variety of ways using Imaginator's unique and highly versatile set of display commands. Both novices and experts alike will appreciate Imaginator's extremely complete manual. The tutorial section of the manual not only teaches you how to use Imaginator; it also provides an excellent introduction to the subject to three dimensional computer graphics.

Imaginator runs on Apple® II, Apple II+ and the super new IIe computers (48K required). Works directly with Epson printers with Graftrax. Compatible with many screen dump utilities and graphics screen enhancers.

IMAGINATOR I An entry level program designed to provide the user with all the basic capabilities for creating, manipulating and displaying three dimensional objects. Perspective or orthographics views of an object can be displayed on the screen and then dumped to a graphics printer or saved to disk. (Fully upgradeable to Imaginator II for an additional $40). **$79**

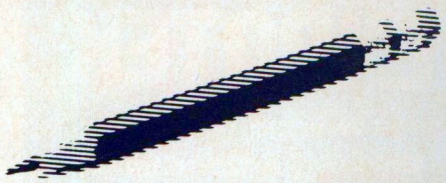
IMAGINATOR II A professional level program allowing the user to create scenes with multiple objects. Makes full use of Imaginator's powerful graphics editor enabling the user to manipulate individual objects in a scene. Perspective or orthographic views of a scene can be displayed on the screen and then dumped to a graphic printer, saved to disk or plotted directly on a plotter. (Available third Quarter 1983). **$119**

Imaginator is a trademark of Shelter Research Institute, Inc. Apple is a registered trademark of Apple Computer, Inc. Epson & Graftrax are trademarks of Epson America, Inc.

TOWNSEND MICROWARE
A DIVISION OF SHELTER RESEARCH INSTITUTE, INC.
921 WATER STREET, P.O. BOX 1200, DEPT. 1008
PORT TOWNSEND, WASH. 98368. 206-385-4621

Circle 26 on Reader Service Card

WRITE -A.P.P.L.E.



The Readers

Drive Noise

Dear Val,

SEVERAL months ago, I called you about a problem with booting and you asked me to let you know when I found a solution. Briefly it was this:

1) All my old (previous to about six or eight months ago) disks boot just fine.
2) No newly inited disks would boot.
3) All newly indited disks work fine in all respects except for booting.
4) Master create had no effect on the problem.
5) Aptest said everthing was OK.

Among my acquaintances, the hardware oriented people were sure it was a software problem and the software oriented people were sure it was a hardware problem. You referred me to Bill Steinberg. He was very gracious and helpful. I sent him three diskettes and he pronounced DOS in good order. The problem persisted.

I wrote a program to read any sector, up to one track at a time, into a known location, repeat with a second disk at a different location and compare the two. This disclosed no differences anywhere on tracks 0, 1 or 2 between the disks which would boot and those which wouldn't boot. Steinberg was right.

Now the good news. I met Randy Selman (Business Computer Consultants, P.O. Box 7001 Boca Raton, Fl. 33431). He diagnosed the problem as loss of sensitivity in the head of the disk drive such that it was picking up noise from the analog board directly above it. He made an aluminum shield to fit under the analog board. Now I can, once more, init diskettes which will boot. Selman says this is a temporary fix which may last indefinitely. If the problem recurs, a new head is the answer. So far, after more than two months, all is well.

Herbert C. Minard
6900 N.W. 14 Court
Margate, FL 33063

Drawing Inverse Borders

Editor:

An easy way of drawing lines without graphics mode is using INVERSE function and FOR-NEXT loops. Need to double space between inverse lines.

This can be used to separate text on a full screen (easier to read) or used to draw lines to graph variables.

Example pre-determined lines:

```
10  HOME
20  INVERSE
30  FOR C = 1 TO 20:
      REM
HORIZ LINE
40  VTAB 14: HTAB C:
      PRINT " ": NEXT
      C
50  FOR R = 6 TO 22:
      REM
VERT LINE
60  VTAB R: HTAB 20:
      PRINT " ": NEXT
      R
70  NORMAL
```

Example Horizontal variable line:

```
10  HOME
20  VTAB 9: HTAB 1:
      PRINT "NAME X"
30 S = 5: REM
SCALE OF 1:
COL = 5 (100 / # OF
COLS WANTED IN GRAPH)
40 P = 65: REM
P CALCULATED AS 65%
50 G = 10 + (P / 5):
      REM
START COL 10, ADD
CALCULATED AMT
DIVIDED BY SCALE
60  INVERSE
70  FOR C = 10 TO G
80  VTAB 9: HTAB C:
      PRINT " ": NEXT
      C
90  NORMAL
100  VTAB 9: HTAB G:
      PRINT P"%"
```

B. J. Wilson
P.O. Box 12628
Seattle, WA 98111

Pascal Prefix Suffix

Dear Val,

Mr. Walker, in his letter on "POKEing Pascal Prefix" in the January issue of *Call -A.P.P.L.E.* states that Pascal 1.1 will only set the Prefix to a volume name, not a device. He has also presented a Pascal program which will do this as well as some other things.

There is a very simple way to get the Prefix to a volume number as described in the *Apple Pascal Operating System Reference Manual Addendum* which came with version 1.1(page II—ED). This is also true of *Softech IV.0.* See page 69 of *The Users Manual—Pascal ED).* This feature allows one to change diskettes in a given drive without having to change the Prefix each time a different diskette is placed in that drive. To use the Prefix command in this way, the Prefix must be set when no diskette is placed in the drive that you wish to designate as the Prefix disk drive. If, when you are setting the Prefix, a diskette is in the drive that is designated as the Prefix disk drive, the Prefix will be set to the name of the diskette in that drive. This is what Mr. Walker was trying to avoid by using his program.

Thus all that is required is to put the boot diskette, Apple1, in the boot drive and no diskette in Volume #5, enter the filer and type P for Prefix. When it asks what volume type #5. Now typing for example L followed by an asterisk will list boot diskette.

Of course Mr. Walker's program performs some additional functions, but is not needed if just the Prefix needs changing.

Sutton Redfern, Ph.D.
Panorama City
189 Circle Drive
Lacey, WA 98503

Wordstar Inverse Video

Dear Val,

In the January issue of *Call -A.P.P.L.E.* there was an Apple Quickie which told how to set up Wordstar inverse video on the help screen and block moves. The information was not correct and was not presented properly. The correct way to install the patches is to set (hex) locations as follows:

Location	New Value
0284	02
0285	1B
0826	28
028B	02
028C	1B
028D	29

This will not only set the help screen to inverse and cause block markers to be in inverse chip on the Videx board.

I hope this will help the membership.

Mike Salesin
5123 Corners Dr.
W. Bloomfield, MI 48033

EXPOTEK

2723 W. Windrose • Suite 3
Phoenix, Arizona 85029

1-800-528-8960

GUARANTEED LOW PRICES

APPLE CARDS
16K RAM — $78 Z80 CARD — $245
Videx Card — $227 Smart Term — $279
Micro Soft Prem. PK — $498

ALTOS
5-15D — $2249 5-5D — $4399

ANADEX
9500A — $1290 9501A — $1290

ATARI
400-16K — $289 800-48K — $645
410 — $77 810 — $429
830 — $155 850 — $159

CITOH
F-10P — $1370 F-10S — $1375
1550P — $659 Prowriter 1550CD — $709
8510AP — $479 8510ACD — $570

DATASOUTH
DS120 — $595 DS180 — $1175

DIABLO
630RO — $1945 620 — $1099

APPLE-LOOK-A-LIKE Call

HAZELTINE
1500 — $995 ESPRIT — $498

MICRO SCI A2
$300 — Apple Drive/Card — $375

MODEMS
HAYS — MICROMODEM — $285
HAYS — SMARTMODEM — $219
HAYS — 1200 Baud — $530

MONITORS
Amdek — Color II RGB — $649
Amdek 300 — $140 Color I — $300
BMC Green — $88 USI Amber — $159

NORTHSTAR
Advantage — $2600 Horizon 64QD — $2600

NEC
8023 — $469 7710 — $2220

OKIDATA
M80 — $305 M83A — $649
M82A — $399 w/Tractor & Grap. $479
M84P — $989 M84S — $1090

TELEVIDEO
802 — $2599 802H — $4450
806 — $4950 800A — $1290

TELEVIDEO
910 — $569 925 — $718
920 — $735 950 — $899

TI
810 — $1240 820 — $1795

ZENITH
Z19 — $689 Z89 — $2129

SOFTWARE
All Major Brands — $CALL

DISKETTES/BOXES
Elephant — $20 Scotch — $25 Dysan — $35

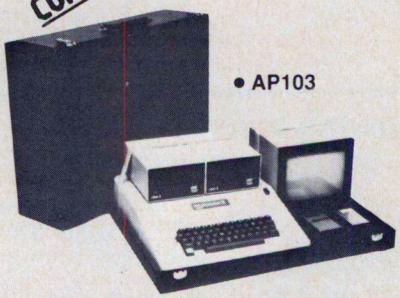
All Prices Subject To Change

Customer Service
602-863-0759

Circle 29 on Reader Service Card

---

computer case company



• AP103

Attache-style cases for carrying and protecting your complete computer set-up. Accommodates equipment in a fully operational configuration. Never a need to remove equipment from case. Simply remove lid, connect power, and operate.

AP101	Apple II with Single Drive	$109
AP102	Apple II with Two Disk Drives	119
AP103	Apple II, 9 Inch Monitor & Two Drives	129
AP104	Apple III, Two Drives & Silentype Printer	139
AP105	13" Monitor with Accessories	99
AP106	AMDEK Color Monitor	119
RS201	TRS-80 Model I, Expansion Unit & Drives	109
RS204	TRS-80 Model III	129
AT301	ATARI Computers with Peripherals	109
P402	Centronics 730/737 & Radio Shack Printer	89
P403	Epson MX70/80 or Microline 82A	89
P404	Epson MX100 Printer	99
P405	IDS 560 or Prism 132 Printer	109
P406	Starwriter/Printmaster F-10 Printer	119
P407	Okidata Microline 83A or 84 Printer	99
P408	Prowriter 2 Printer	99
P409	Prowriter (Apple Dot Matrix) Printer	89
IB501	IBM Personal Computer	129
IB502	IBM Monitor	99
HP601	HP41 with Accessories	99
CM703	Commodore Model 64 with Drives	119
CM704	Commodore Model 64 with Dataset	109
NS010	North Star Advantage	139
CC80	Matching Attache Case (5")	85
CC90	Matching Attache Case (3")	75
CC91	Matching Accessory Case	95
CC92	5.25" Diskette Case	49

computer case company

5650 Indian Mound Court
Columbus, Ohio 43213
(614) 868-9464

CALL TOLL FREE
800-848-7548

Circle 31 on Reader Service Card

BYTES & PIECES

Extra Special!
D.B. Master
$151.95

All Programs On Disk

LIST PRICE $29.95 • OUR PRICE $22.16

Spider Raid	Canyon Climber
Lunar Leeper	Castle Wolfenstein
Pest Patrol	Spy's Demise
Succession	Free Fall
Warp Destroyer	Sea Fox
Beer Run	Hard Hat Noah
Apple Panic	Escape from Rungistan

LIST PRICE $34.95 • OUR PRICE $25.86

Star Maze	Kabul Spy
Zargs	Deadly Secrets
Bandits	Transylvania
Cannonball Blitz	Crisis Mountain
Choplifter	Mask of the Sun
Hi-Res #4	Knight of Diamonds
Frogger	

LIST PRICE $39.95 • OUR PRICE $29.56

Zork 1, 2, or 3	S.E.U.I.S.
Star Cross	Cytron Masters
Type Attack	Galactic Gladiators
Blade of Blackpool	Ultima
Rendezvous	Aztec Adventure

	LIST	OURS
Arcade Machine	59.95	44.36
Bag of Tricks	39.95	29.56
D.B. Utility Pak 1 or 2	99.00	73.26
Deadline	49.95	36.96
Executive Secretary	250.00	184.96
General Manager 2.0	229.95	170.16
Graphics Magician	59.95	44.36
Home Accountant	74.95	55.46
Lisa Ed. System	119.95	88.76
Logo (Terrapin)	149.95	119.96
Maxell Disks (10)	55.00	31.00
Multiplan	275.00	203.49
Nibbles Away II	69.95	51.76
Screenwriter II	129.95	96.16
Screenwriter Pro	199.95	147.96
Snooper Troops 1 or 2	44.95	33.26
Super Text 40/56/70	125.00	92.49
System Saver Fan	89.95	76.46
Tasc Compiler	175.00	129.49
Ultima II	59.95	44.36
Utility City	29.50	21.83
Visicalc	250.00	199.96
Voice Box	215.00	159.09
Wildcard	129.95	116.96
Wizardry	49.95	36.96
Zoom Grafix	39.95	29.56
SPECIALS		
Crossword Magic	49.95	34.96
D.B. Stat Pak	99.00	71.26
Graforth	75.00	54.49
Replay Card	129.95	108.96
Sensible Speller	125.00	90.49
Time Zone	99.95	71.96

COD • Money Orders • Certified Checks • Personal Check Allow 2 Weeks
NYS Res Add 7.25% Sales Tax • Orders Under $100.00 Add P&H
$2.00 U.S. $2.50 Canada, U.S. Funds
Foreign Charges Only Min P&H $6.00

• **SEND FOR FREE PRICE LIST #900** •
Source TCP637

BYTES & PIECES (516)751-2535

Box 525 Dept. C • E. Setauket, N.Y. 11733

Circle 30 on Reader Service Card

FIXAPCPM

Murray Arnow/Call —A.P.P.L.E. Consultant

```
;-----------------------------------------;
;                FIXAPCPM                 ;
;               Version 1.0               ;
;               by M.Arnow                ;
;              January,1983               ;
;-----------------------------------------;

;This program corrects the problem of
;clearing the screen and homing the
;cursor when the Apple IIe has an
;Apple 80 column card in the aux.
;connector slot.

;The problem derives from Microsoft's
;CP/M BIOS initializing the peripheral
;cards in all the slots on a warm boot.

;The solution is to remove the initial-
;ization routine from the warm boot
;and to make the CONTROL-RESET perform
;a cold boot. The cold boot initializes
;the peripheral cards.

;This program will only work for CP/M
;versions 56K (2.20B) and 60K (2.23).

        ORG     100H

BOOT    EQU     0
BDOS    EQU     5
LF      EQU     10              ;line feed
CR      EQU     13              ;carriage return
FUNNY   EQU     0F3F4H          ;Monitor funny byte

TRUE    EQU     1
FALSE   EQU     0

;Choose CP/M version here.

CPM56   EQU     TRUE
CPM60   EQU     FALSE

;Load disk I/O routines.

        LHLD    BOOT+1          ;get wrm-boot addr.
        LXI     D,27            ;load SETTRK offset
        DAD     D               ;HL points to SETTRK
        LXI     D,SETTRK        ;DE points to local
                                ;SETTRK
        LXI     B,15            ;set counter
        DB      0EDH,0B0H       ;Z80 op-code for LDIR
                                ;to move routines for
                                ;local use

;Make CONTROL-RESET do a cold boot.

        MVI     A,0
        STA     FUNNY

;Fix warm boot currently in RAM.

        IF      CPM56
INITJMP EQU     0DAD8H          ;where the warm boot
                                ;calls for the init.
        ENDIF

        IF      CPM60
INITJMP EQU     0FAC4H
        ENDIF

        STA     INITJMP         ;store NOP's here
        STA     INITJMP+1
        STA     INITJMP+2

;Print header and instructions.

        LXI     D,HEADER
        MVI     C,9
        CALL    BDOS

;Get response and proceed.

        MVI     C,1
        CALL    BDOS
        CPI     'Y'
        JZ      PATCH
        CPI     'y'
        JZ      PATCH
        LXI     D,NPATCH        ;print exit mssg.
        MVI     C,9
        CALL    BDOS
        JMP     BOOT            ;return to system
                                ;if no 'Y' or 'y'

;Patch disk in DRIVE A: with corrected BIOS.

PATCH:  MVI     C,0DH           ;reset disk system
        CALL    BDOS
        LXI     B,BUFFER        ;set READ/WRITE buffer
        CALL    SETDMA
        LXI     B,0             ;set TRACK=0
        CALL    SETTRK
        LXI     B,0FH           ;set SECTOR=5
        CALL    SETSEC
        CALL    READ            ;read sector
        MVI     A,0             ;change funny byte
        STA     BUFFER+74H

        LXI     B,0FH           ;reset sector
        CALL    SETSEC
        CALL    WRITE           ;write the change
                                ;to the disk

        IF      CPM56
        LXI     B,2             ;set TRACK=2
        CALL    SETTRK
        LXI     B,0DH           ;set SECTOR=0EH
        CALL    SETSEC
        CALL    READ            ;read sector
        MVI     A,0             ;remove call to
        STA     BUFFER+58H      ;peripheral card
        STA     BUFFER+59H      ;initialization
        STA     BUFFER+5AH      ;routine
        LXI     B,0DH           ;reset sector
        CALL    SETSEC
        CALL    WRITE           ;write the change
                                ;to the disk
        LXI     D,YPATCH        ;print exit mssg.
        MVI     C,9
        CALL    BDOS
        JMP     BOOT            ;return to system
        ENDIF

        IF      CPM60
        LXI     B,2             ;set TRACK=2
        CALL    SETTRK
        LXI     B,5             ;set SECTOR=0CH
        CALL    SETSEC
        CALL    READ            ;read sector
```

A **Call —A.P.P.L.E.** QUICKIE

```
]LIST

1  REM

-- RESET TO SLOT 3 --

  BY W.J. CHEESEMAN

   CALL -A.P.P.L.E.

10  REM
This routine points the
reset vector ($3F2) to $300;
loads at $300 a short routine
that sets CSWL, CSWH to point to
slot 3, reconnects DOS
and warmstarts Applesoft.

11  REM
Thus, pressing [RESET] will
leave an 80-column card active,
instead of returning to the
Apple's internal keyboard and
video display (see Apple ][
Reference Manual, p. 36).

12  REM
This routine uses page 0 and
page 3 pointers only, so it can
be used with any size Apple.

20  PR# 3
30 A$ = "300:A9 00 85 36 A9 C3 85
      37 20 EA 03 4C 00 00 N 3F2:
     00 03 A6 N D9C6G": FOR I = 1
      TO  LEN (A$): POKE 511 + I,
      ASC ( MID$ (A$,I,1)) + 128:
      NEXT : POKE 72,0: CALL  - 1
     44
40  PRINT "PRESS RESET AND SEE WH
     AT HAPPENS!"
```

FIXAPCPM

```
        MVI     A,0             ;remove call to
        STA     BUFFER+44H      ;peripheral card
        STA     BUFFER+45H      ;initialization
        STA     BUFFER+46H      ;routine
        LXI     B,5             ;reset sector
        CALL    SETSEC
        CALL    WRITE           ;write the change
                                ;to the disk
        LXI     D,YPATCH        ;print exit mssg.
        MVI     C,9
        CALL    BDOS
        JMP     BOOT            ;return to system
        ENDIF
HEADER: DB      CR,LF,LF
        DB      '*******************************',CR,LF
        DB      '*                             *',CR,LF
        DB      '*          FIXAPCPM           *',CR,LF
        DB      '*                             *',CR,LF

        IF      CPM56
        DB      '*          56K Version        *',CR,LF
        ENDIF

        IF      CPM60
        DB      '*          60K Version        *',CR,LF
        ENDIF
        DB      '*                             *',CR,LF
        DB      '*          by M.Arnow         *',CR,LF
        DB      '*                             *',CR,LF
        DB      '*        January,1983         *',CR,LF
        DB      '*******************************',CR,LF
        DB      CR,LF
        DB      'Do you wish to patch FIXAPCPM',CR,LF
        DB      'to the CP/M BIOS on the disk',CR,LF
        DB      'in DRIVE A: (Y/N) ? $'

NPATCH: DB      CR,LF,LF,'The disk has been left '
        DB      'unaltered.',CR,LF,'$'

YPATCH: DB      CR,LF,LF,'The disk has been '
        DB      'updated.',CR,LF,'$'

SETTRK: DS      3               ;track in BC
SETSEC: DS      3               ;sector in BC
SETDMA: DS      3               ;R/W buffer addr. in BC
READ:   DS      3               ;read sector to BUFFER
WRITE:  DS      3               ;write BUFFER to sector
BUFFER  EQU     $               ;READ/WRITE buffer

        END
```


BEEP! LOST YOUR PROGRAM?

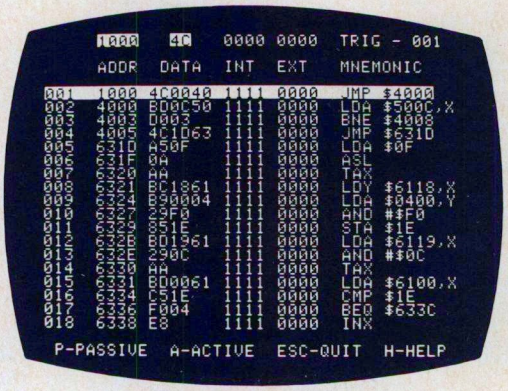
BUS RIDER
LOGIC ANALYZER FOR THE APPLE II

The Bus Rider is a self diagnostic development tool that allows real time analysis of software and hardware in the Apple II computer.

The Bus Rider provides:

- Monitors and saves 512 cycles of the address and data bus, NMI, IRQ, DMA, R/W and 4 external lines.
- Pretrigger viewing of up to 512 samples.
- 4 external inputs with variable threshold reference.
- Display cycle by cycle execution or 6502 disassembled code.

The Bus Rider comes complete with Bus Rider circuit card, reference manual, Bus Rider software diskette, and 10 easy hook external input cable. The total system price is **$395.00**



Bus Rider – Disassembled Display

RC Electronics Inc.
5386 Hollister Avenue, #D
Santa Barbara, CA 93111
(805) 968-6614 • TELEX 295281

MC Visa

Circle 14 on Reader Service Card

THE WORD

(Though not necessarily from above)

Mike Shadick

"THE word is dead," some said. They even came up with a new name for the posthumous era.

They called it the *post–literate* age. (An interesting choice of *words*?)

"They" claimed that the new age, which was upon us, was one in which words and the printed word in particular, was fading from popular view.

Not, mind you, that anything was *happening* to all of these words. Heaven forbid! It's just that that folks were no longer *reading* the words.

Can you guess what they were watching instead?

So the sages surmised that literacy had (already) reached its Zenith, and that the printed word was, to put it bluntly, on its way out.

People were buying fewer books and newspapers, adding fuel to the sages' fire. Some of the latter, indeed, were going out of business in a basket! People were turning to the electronic media for the stimulus previously supplied by the printed word.

Yes, that's where a lot of the words seemed to be headed.

Out of print.

Their competition was even beginning to be called *video*. Moreover, videophiles (who only a few years ago had been termed boob-tubers) were starting to relate to video in much the same way that their predecessors, and even they themselves, had once related to sources of the printed word. Like libraries.

Yes, friends, it *did* appear that the word was dead. Or would soon be so. Just as "they" had said.

But just about then, something happened.

Something that precious few of "them" had been prepared for. Indeed, IT was an exigency which few, if any, critics of the post–literate era, or of any other era for that matter, had anticipated. Why, even Aldous L. Huxley's *Brave New World* never envisioned IT. Nor had George Orwell, in *Animal Farm* or his infamous Big Brotherism of *Nineteen Eighty–Four.*

What then, was IT?

What else but — the microcomputer.

Today, it has already changed many of our lives. Perhaps of even greater significance, as history will doubtless record, is the fact that the microcomputer is giving the *word* not merely a new lease on life, but a new reason for *living.*

Prior to the advent of the computer in general and the development of the microcomputer in particular, the printed word was a more or less *passive* phenomenon to everyone except the relatively few who created the words — writers, publishers, and such. "They" had the full and complete power over the word. The masses just *read* them, possessing little or no opportunity to *alter* the the words they were reading.

In other words, the people had virtually no power over the printed words. The writers and publishers, *et. al.,* had it *all.*

Then, rather suddenly, IT came on the scene. The microcomputer gave *everyone* the potential to not just *read* the words which were set before them, but to *change* them.

And they called it word processing. It had theoretically been available for quite a while — for "their" use.

But now is was *ours,* as well. It was, indeed, *everyone's.*

Simply one result, we find that today, as we approach the dawn of the Third Millenium, A.D., the rebirth of the word has — as Karen and her Carpenter brother might say — *only just begun.*

Of course, not everyone, everywhere, is as yet into word processing! Nor will everyone *ever* be — including many microcomputerists.

The important thing, however, is that it's *there.* It's accessible, available and obtainable. Not just by "them," but by *everyone.*

Only a few short years ago, such was far, far from the case? Indeed! That was in fact just about the time when "they" were lamenting America's entry into the post–literate era.

Well, what do you know. That was about the briefest, shortest–lived "era" in American history. One which is now little more than a fading memory. For today, we're entering (some would say we've entered it already!) an age in which literacy is being nothing less than reborn. Being re–*created,* if you will.

It is, for a fact, the age of *computer* literacy.

No, I take that back. 'Tis the age of *microcomputer* literacy. And the word has never been more accessible (processable), by so many millions of Americans.

"The word is dying," some were sighing? Well, they're singing no such tune today.

Heck, they're too busy at the keyboards of their word processors!

Today, thanks to the microcomputer, the word isn't dead, or dying, or even just surviving. It's *thriving.*

And that, my microfriend, *computes!*

A.P.P.L.E. Members Only

Ap.Graphics
The Filler

A set of machine language routines that quickly fill Hi-Res areas with colors or patterns.

The Filler offers:

- Picture/Pattern Editor
- Ampersand Vectored
- Extremely Fast
- 21 Colors with Pre-defined Patterns

Member Price
$22.00

A.P.P.L.E. Orders
21246 68th Ave. S.
Kent, WA 98032
(206) 872-2245

Washington state residents add 6.8% sales tax
Foreign members add $6.00 for shipping/handling
Canadian postage being revised

BEEP: A TALE OF (T) ERROR

Christopher Volpe

BACK in the days of the late Integer Basic, one might somewhere along the line use an arithmetic operation with more than twelve sets of nested parentheses. He would then hit [Return] and the computer would reply with a "TOO LONG ERROR (beep!)" message. Realizing that the line he had typed was less than 128 characters long, he would then look in the back of the *Apple II BASIC Programming Manual* in the error message section and see that the error message can occur also when using too many parentheses.

Although I have never needed to use more than twelve nested parentheses, I have always wondered why they couldn't have made another more appropriate error message and spared us the confusion.

One day I found out what was wrong. I was using a BASIC program that was originally used to look at the DOS error message and command tables to look through Integer Basic and see what I could find. The program, after printing a long series of garbage characters, came to and displayed the whole list of Integer Basic error messages. (To see this list, type in the machine language program in Listing 1 and run it. The result should be the list of messages in Table 1. Note: To run this program you must have Integer Basic up.) Look at the message towards the end of the first line. It reads "TOO MANY PARENS." It's right there in the ROMs and always has been for the past five years or so but was just never shown either by the computer or in the manual. This is the error message that you should get but don't because of a bug in Integer Basic. (Maybe The WOZ made a typo?)

The error is in location $E47A in the Integer ROMs. The value stored there is $06 which corresponds to location $EB06, the beginning of the "TOO LONG" error message. It should be $1C which corresponds to location $EB1C, the beginning of the "TOO MANY PARENS" error message.

Correcting this bug is very simple with a RAM card, more difficult with an EPROM programmer, and quite impossible without either of the two. And since I don't have an EPROM programmer, I'll stick with the RAM card method. To make the correction using a RAM card on an Apple II plus, complete the following steps:

1. Enter the monitor with "CALL -151 [Return]"
2. Type "C081 C081 [Return]" to select motherboard ROM read and RAM card write enable.
3. Insert your System Master disk and type "BLOAD INTBASIC, A$D000 [Return]"
4. Type "E47A:1C [Return]"

Now, to test the correction, type "PRINT ((((((((((((((0)))))))))))))" and hit "[Return]." If everything is working the way it should (in other words if I didn't mess something up), the computer should respond with "***TOO MANY PARENS ERR (beep!)." Now you may want to save the corrected version of Integer Basic to your disk (not your write-protected SYSTEM MASTER) by typing "BSAVE INTBASIC,A$D000, L$2FFF".

I'd like to thank Bert Kersey for his program "Cruising Through DOS" which helped me find the Integer error message table.

Listing 1

```
*300L

0300-   AD 06 EB    LDA   $EB06
0303-   20 ED FD    JSR   $FDED
0306-   EE 01 03    INC   $0301
0309-   AE 01 03    LDX   $0301
030C-   E0 99       CPX   #$99
030E-   F0 03       BEQ   $0313
0310-   4C 00 03    JMP   $0300
0313-   A9 06       LDA   #$06
0315-   8D 01 03    STA   $0301
0318-   60          RTS

031A-   00          BRK
031B-   00          BRK
031C-   00          BRK
031D-   00          BRK
031E-   06 29       ASL   $29
0320-   3F          ???
0321-   69 20       ADC   #$20
0323-   49 E0       EOR   #$E0
0325-   48          PHA
```

table 1

Integer Basic error message table

TOO LONGSYNTAXMEM FULLTOO MANY PARENS—STRINGNO ENDBAD BRANCH16 GOSUBSBAD RETURN 16 FORSBAD NEXTSTOPPED AT ★★★ ERRM>255RANGE DIMSTR OVFL\MRETYPE LINE

ROBOTICS SERIES

Michael Thyng/Call —A.P.P.L.E. Staff Writer

3) The Design and Where to Find the Building Materials

FIRST, I need to make some comments to those who have written in. Thanks for your encouragement. It helps. For the record, and for any readers who may be confused about the dollars involved in this project: I am building this robot with my own designs, and with my own money. It is *not* an *A.P.P.L.E.* product. I am showing you what I am doing through these articles so that you can make comments and possibly follow along, building as we go. If you choose to make a robot just like the one I am making — wonderful. You make yours, I'll make mine. Hopefully, we will be able to help each other. The input I have requested from you is on a purely voluntary basis. If you want to contribute ideas (and believe me, many have), good! If you want to do some circuit design and/or construction, great. But there are no funds to pay for such a thing. The reason for the article is to show you, step by step, how you might go about building a robot. In a more positive vein, if you have a question about something, or you find a mistake or have a question about something you've seen done in the project, write in. I have tried to modify the designs to reflect your good ideas.

I have not been able to respond with letters to most of you who have written. Let me try to answer you here. Now, for those who have suggested the many ideas and asked the myriad of questions, yes, I have thought of some kind of voice. No, it isn't going to be one of the first things implemented. Yes, you will need to be able to read circuit diagrams. I will try to offer step-by-step construction hints and tell what each component looks like. No, you won't need to be able to design circuits. Yes, the instructions are *intended* to be quite explicit. Whether they are for you or not, well, only time will tell. The robot will be approximately two feet tall, cylindrical, and move about on wheels. The only component which may cause an on-going problem is your availability of a good, cheap microcomputer-on-a-board. I am going to use the wonderful KIM-1. But for those of you who need to buy a micro to dedicate to a robot, I might suggest the SINCLAIR. I prefer sticking with 6502, but I'm sure that many of you may try some other types of processors. There are a lot of choices out there.

If you come up with something different from what I am writing about, send it in. There are a lot of "hardware" types out there who read this magazine. Feel free to write and suggest a processor that you have had success with in this dedicated controller-type application. Some of you suggessed that I use the Apple board. I didn't consider that because 1) I didn't have a Peripheral Interface Adapter chip on my board (weak excuse) and 2) I don't know where to get a "bare-bones or nearly bare-bones" Apple][. Some of you were concerned about the software. This is the truly unique part of each robot. For the start of the project, the only control we install will be like the Milton Bradley BIG TRAK. That is, we will tell it which direction and how far to go up to some number of steps.

Some might ask why we would need to know where to get the materials before we complete the physical design. Being a pragmatic type, I look to the proverbial "bottom-line." In our robot, the bottom line is that if you build to meet specifications the cost could be astronomical. I don't know what horsepower motor we need to make it go. If I design my robot to be powered by a motor that needs to fit into a certain size space, then I *must* find that size motor. With my luck, the size I would have would be a unique or short run design that would cost much more than a motor just a little bit larger or a little bit smaller. Moral: know what you're doing *or* be flexible. (I have to settle for being flexible.) The design we're going to use will be vaguely reminiscient of a certain Droid called R2-D2. While I would like to be creative and design a unique robot that would impress my friends and catch the eye of Steven Spielberg, I'm willing to settle for less. I plan to construct something that looks like an inverted garbage can on wheels. See figure 1.

Notice that the arm has not been incorporated. Let's take first things first. We'll get it moving first, then moving under control, *then* we'll worry about having the robot pick things up. Also, notice that there are three wheels and two motors. I think that reliablility is of utmost importance. Therefore, I used two motors, each driving a separate wheel. The third wheel is just a caster that provides stability. The batteries are 12 volt motorcycle batteries. I'm using two to provide electrical isolation between the high draw 12 volt motors and the relatively low draw electronics of the microcomputer. I'll describe the relays when we get to the drive train. The electronic shelf is a two foot diameter 1/8th inch piece of masonite. It was just something I had in the house. A piece of plastic or possibly aluminum plate would work just as well.

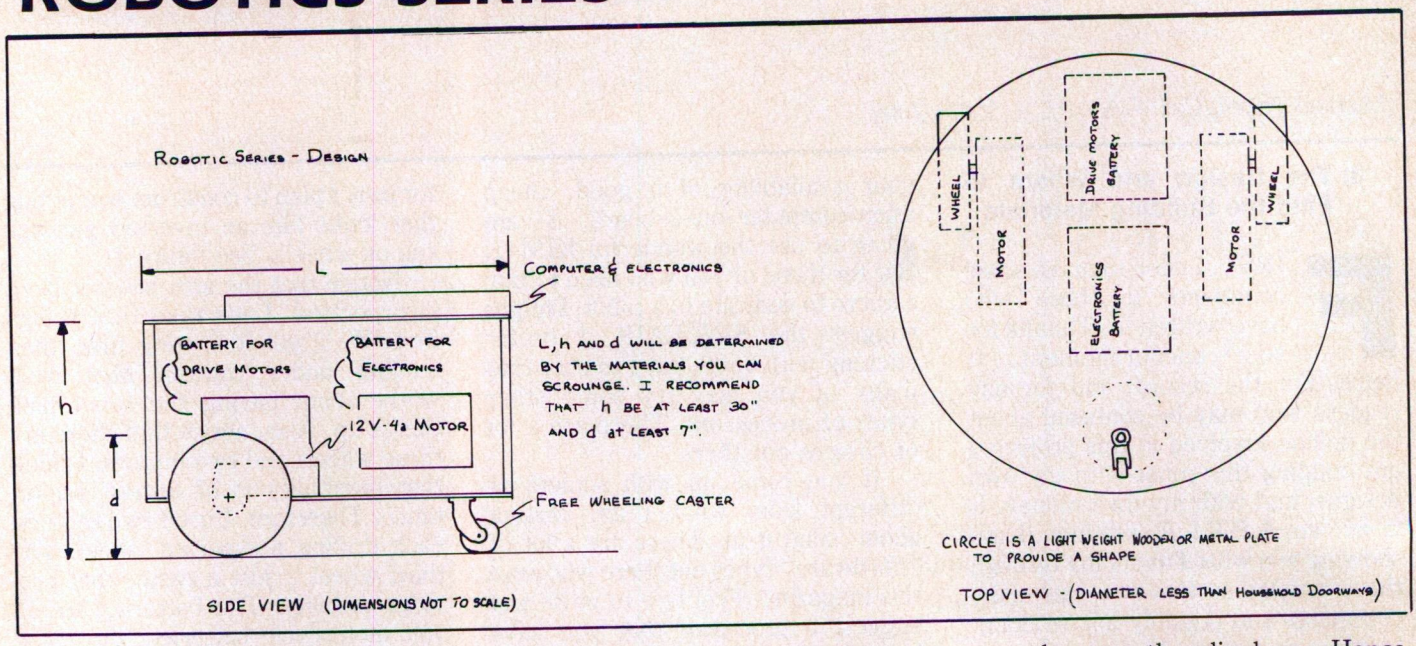
O.K. back to materials. Overall, use aluminum structural components. Aluminum offers a fine functionality to price ratio. It is relatively light, has plenty of strength for our needs, and can be purchased from surplus materials stores. I recommend "L" shaped and box shaped extrusions for framing and open construction.

I got lucky. A friend, who thinks that a home made robot is a good idea, gave me two matching 12 volt electric motors. They are the electric window motors from a 59 Dodge. Another robot builder in my area used Lincoln Continental electric window motors. He paid $10 each at a local wrecking yard. They would have cost $20 each, but he, instead of wrecking yard personnel, removed the motors from the car.

Let's discuss the electric motors a bit more. I tested the ones I got. They draw four amps. Four amps! That is a lot!! And it creates two major problems. First, it means a big draw on your batteries. I can solve that by using about a 20 amp-hour motorcycle battery. That will give the robot a travel time of about a full

—●→

ROBOTICS SERIES



two hours. I don't know what the heat build-up might be. Could somebody suggest the answer to that? Second, it means that I have to use more powerful components to control that much power with a microcomputer. More powerful means more expensive. And for this application, it also means less available. There exists such a thing as a solid state relay that will let a computer with its itty-bitty (technical electronic talk) voltage level control a motor with a big voltage level. Unfortunately for me, the four amp requirement is just off the scale of the more common solid state relays. So, one of you called me and suggested the Teledyne 603-22. Great. I went to my local supplier who explained to me that it was currently out of stock. Not great. How long to get it in? 18 weeks. I'll give the the solution when we do the electronics portion of this series. (It is solved as of this writing.)

Where do you go for wheels? I bought mine from the hardware store where the clerks always frown. I weighed the wheels in the nail scale and found something interesting. The 7 inch diameter wheels weigh 1 pound each. The 8 inch diameter wheels weigh 1.75 pounds each. The more weight the motors have to push, the faster the power is used up. I wanted to use the largest size wheel I could without adding too much extra weight to the robot. Where is the robot going to be spending most of its traveling life? In the house — travel-travel across than linoleum. Hence, big diameter. Where else can you go for wheels? You can go back to the surplus store.

Let me give you some words of encouragement. The very hardest part is getting started. If you really want to build your own robot, go now and get two pieces of paper. Draw yourself a picture of what you want it to look like on one sheet. Write down exactly what you want your robot to do on the second. You now have a plan. Now add dimensions to the picture you drew. Go to the local hardware store or surplus store and buy something. Come home and drill some holes. Bind it together with some nut and bolts. There, now you have an investment in technology. Do you want to throw that investment away? Of course not. You are part of the way there. It will go very slowly until you have the wheels on. After that, it will be easy to foresee the end of the project.

Next time — Doing the construction...

If you are interested in contributing ideas to this project, *write* to me at:

Michael Thyng
Robotics Project
6242 - 35th N.E.
Seattle, WA 98115

I will answer all your reasonable questions as quickly as possible either by return mail or through this column.

«DAJAX»

Save Time Anytime.

Software For Your Apple
By Direct
Modem Downloads

Requirements:
Apple II/II+
48 K or more
DOS 3.3
D. C. Hays Micro Modem II

«DAJAX»

CALL 24 HOURS A DAY.
(404) 736-3529
Visa And MasterCard
Only Accepted

Circle 67 on Reader Service Card

JCL PROBLEMS?

• LEARN IBM JCL •

with **INTRODUCTION TO IBM OS/JCL.** This tutorial covers all the basics from syntax to blocking and space allocation to IBM utilities. Don't be intimidated by the wizards, learn their secrets! Institutional version handles up to 100 students concurrently – $50. Individual version for $25. $3 extra for COD.

Applied MicroSystems, Inc.

P. O. Box 832
Roswell, Georgia 30077

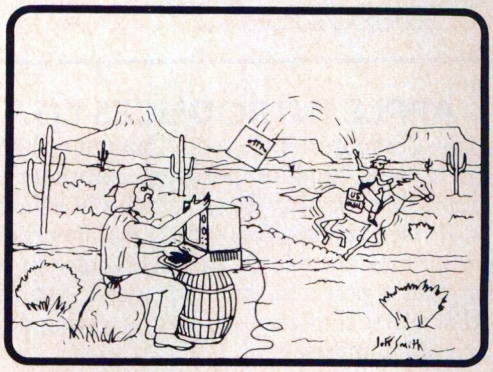
(404) 475–0832

APPLESOFT DOS 3.3
GA residents add 4% sales tax. Order our CATALOG-ONA-DISK for $3.

Circle 68 on Reader Service Card

TELEPHONE SOFTWARE CONNECTION™ INC

No MODEM yet??? We'll find another way to reach you! Send for our free catalog.



Software via Teledelivery™!

1-213-516-9432

POST OFFICE BOX 6548 - TORRANCE, CALIFORNIA 90504

© 1982 Telephone Software Connection, Inc.

Circle 69 on Reader Service Card

A.P.P.L.E. Members Only

Ap.Graphics
The Filler

A set of machine language routines that quickly fill Hi-Res areas with colors or patterns.

The Filler offers:

- Picture/Pattern Editor
- Ampersand Vectored
- Extremely Fast
- 21 Colors with Pre-defined Patterns

Member Price
$22.00

A.P.P.L.E. Orders
21246 68th Ave. S.
Kent, WA 98032
(206) 872-2245

Washington state residents add 6.8% sales tax
Foreign members add $6.00 for shipping/handling
Canadian postage being revised

We Lowered Our Prices
& Expanded Our Product Line

Real Time Game Development Systems

The **frob:** The hardware/software system that allows anyone the opportunity to design home video cartridges using your Apple II® 48K computer.

- ✓ Atari®2600 VCS™ 4K System
- ✓ Atari VCS 16K Bank Switching System
- ✓ Atari 5200-4K, 8K, 12K, & 16K Systems

Watch for our:

- ✓ Stand-alone system allowing development on the Atari 400/800, IBM® PC, VIC-20®, TRS-80®& more *for your Atari* game console
- ✓ Colecovision™ System

The Miracle of Creation Can Be Yours™

™

Call Now To Order
408-429-1552
We accept all major credit cards

For more information, write to:
frobco, A Div. of Tri-comp Polytechnical
P.O. Box 8378, Santa Cruz, CA 95061-83078

Circle 70 on Reader Service Card

TOMORROW'S APPLES TODAY

Kevin O. Donohoe/Editor

APPLE INTRODUCES LISA™

Apple Computer, Inc. today announced the Lisa™ system, a new kind of personal computer aimed at revolutionizing the way work is done in office environments. Priced under $10,000, Lisa incorporates the latest software and hardware technology in a desktop system that's powerful and advanced, yet so easy to learn that a first-time user can begin putting it to work in less than 30 minutes.

Lisa™, a personal computer for the office, was introduced by Apple Computer, Inc., in January, 1983. The 32/16-bit computer, aimed at revolutionizing the way work is done in the office environment, features six integrated software applications that cover the core functions of today's office. Lisa incorporates the latest software and hardware technology in a system that's powerful and advanced, yet is so easy to learn that a first-time user can put it to work in less than 30 minutes.

Lisa (Local Integrated Software Architecture) forges a new relationship between users and computers, allowing people to work in a more natural way without having to adopt rigid computer conventions or special languages. Lisa's screen displays simple pictures of documents, folders, and other familiar things in a typical office. A palm-sized device called a "mouse" is used to point to and manipulate these items, and to perform desired tasks.

Six integrated software applications for Lisa cover the core functions of today's office: spreadsheet analysis, word processing, business graphics, graphics design, personal filing, and project management. The user is able to view multiple applications and documents on the screen simultaneously, transferring information between them quickly and easily.

The Lisa Environment

Lisa fosters a more productive, efficient office environment by following the work habits of the individual. The result is a system that functions as a natural extension of its user.

Largely responsible for Lisa's ease of use are pictures on the screen of objects usually found on an office desktop, and the mouse, which moves the screen's pointer. The user controls the Lisa system intuitively by pointing at and selecting the symbol for a file folder, a memo pad, a wastebasket, or other familiar objects.

Once selected, an object is used just like its real-life counterpart. A file folder, for example, can be opened and the contents revealed; documents can be refiled, copied, taken out and changed, put in a new file, or thrown away.

The basic functions in all six of Lisa's integrated software applications operate in the same way. Thus, once a person has used one application, he/she can learn the others more quickly.

Transferring information from one application to another also is made easy with Lisa. For example, when writing a cost analysis report, the user can quickly shift from word processing to a spreadsheet model to perform the calculations. At that point the user may choose to "cut out" the spreadsheet and "paste" it into the report, or to transfer the spreadsheet information to a business graphics application and turn out presentation-quality graphs—all at the touch of a button.

Designed for the Office: Six Integrated Software Applications

Lisa's software applications are integrated business tools designed to streamline all the work of the office professional, from report writing and forecasting, to financial modeling, project management, and product design.

LisaCalc™ is an electronic spreadsheet for handling budgets, forecasts, and other row-and-column format models that help users plan better. Different assumptions about a forecast situation can be made, plugged in, and results calculated and displayed instantly. Using the mouse, LisaCalc models can be transferred into other programs in seconds.

LisaWrite™, a word processing program lets the user generate any written text, from a short memo to a lengthy report. Inserting words, rewriting text, and other sophisticated editing functions can be done quickly and easily. With the mouse, the user can easily select bold, italic, and underlined text in any combination and in a variety of typestyles—all visible on screen. In addition, a user can print one document while simultaneously creating or editing others.

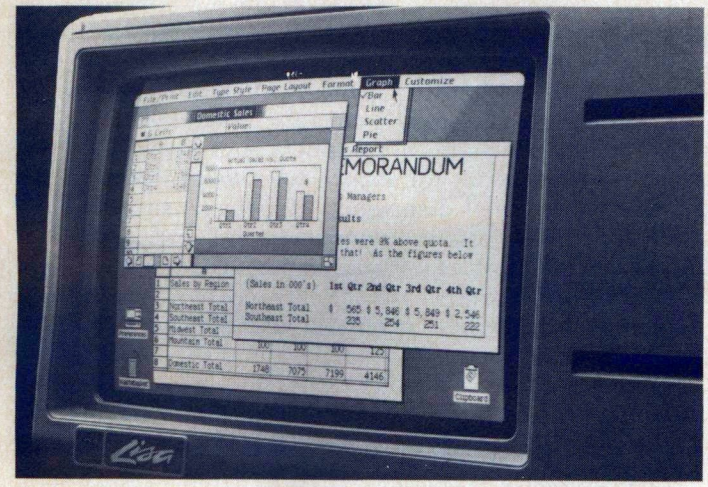
LisaGraph™ allows users to display spreadsheet data and a related graph—either bar, line, bar/line mixture, scatter graph, or pie chart—together on the screen. The program plots up to seven data sets on a single graph. Blocks of data transferred from LisaCalc to LisaGraph are plotted automatically. Graphs can be customized by cutting and pasting them into LisaDraw.

LisaDraw™ enables users to enhance reports and presentations with clear, easy-to-prepare graphics. Using the mouse and a "palette" of shapes that include lines, rectangles, circles, and freehand shapes, the user can draw organization and flow charts, technical diagrams, schematics, and other visual aids. Shapes can be

moved, duplicated, reduced, and expanded automatically to meet the user's needs. Printouts reflect exactly what is on the screen.

LisaList™ is a data base application that searches, sorts, and extracts information quickly and easily. All types of files, from client lists and job candidate data, to competitive product information and other records, can be created and accessed with the program. Sensitive data can be suppressed for security. And columns can be added, deleted, expanded, narrowed, or repositioned at any time without reentering data.

LisaProject™ permits the management of complex projects, including deadlines, resources, and critical paths. One project milestone can be changed and its effect on other milestones will be seen immediately. Results can be printed out in various sizes, from wall charts to single-page reports. Reports are available in Schedule, Task, and Resource formats.



Lisa's™ high resolution screen can display all integrated software applications simultaneously. Shown here are (clockwise, left to right) LisaGraph™, LisaWrite™, and LisaCalc™. Across the top of the screen is the menu bar that lists the options available to start a specific task. After a selection is made, the menu bar's pull down feature displays additional options for completing the task.

Development Tools

Lisa runs BASIC, Pascal, and COBOL high-level languages, allowing users and independent software vendors to augment Lisa's repertoire of applications with programs tailored to specific needs. A Lisa Applications Development Toolkit, to be available later in 1983, will help users and software developers create additional integrated programs for Lisa.

In addition, Lisa will support other major operating systems, including the CP/M™ family, available from Digital Research, and Xenix™, available from Microsoft. Other independent software companies are already developing additional applications.

Lisa's High-Performance Hardware Architecture

Lisa is powered by the MC68000, a 32/16-bit microprocessor that has the capability to combine software, display, keyboard, mouse, and peripherals into the most versatile, easy-to-use system yet developed. The MC68000 features 32-bit internal architecture and a 16-bit external data path. Three other microprocessors control input/output functions, relieving the MC68000 of the additional processing overhead and allowing the system to achieve even higher levels of performance.

The system is configured with one megabyte of main memory and 1.7 megabytes (formatted) of built-in mass storage on two 5 1/4-inch floppy disk drives. These incorporate the Apple 871 disk drive, a high-density, double-sided mechanism which provides greater integrity of data, greater data capacity, and faster data transfer rate than conventional floppy disk drives. Apple's ProFile™, bundled with Lisa, is a self-contained, five-megabyte hard disk storage system. It allows Lisa's applications to be stored on one disk and gives the user the advantage of running all Lisa applications concurrently.

Lisa features a 12-inch, black on white, bit-mapped screen. This high resolution screen displays 364 lines of 720 dots each, letting the user create extraordinary graphics and up to 132 columns and 40 rows of sharp, clear text. Although the mouse replaces special function keys, the keyboard can be programmed for special character functions. Lisa is equipped with a typewriter-style keyboard with a numeric keypad. With a single keystroke, standard characters can be quickly shifted to mathematical symbols or foreign characters.

The system's three built-in interface ports—two serial and one parallel—let Lisa fill a multitude of roles. Using LisaTerminal™, a telecommunication software package being developed by Apple, Lisa functions as a teletype terminal, a DEC VT100 or DEC VT52 terminal, or an IBM-compatible terminal. When linked to AppleNet™, Apple's newly announced local area network, Lisa makes it possible for users to share peripherals and exchange information and files.

Interface to external peripherals is handled through either the parallel port or the two serial ports, one of which is equipped for modem interface. The parallel port is used to support the ProFile. Lisa also provides three additional expansion slots, each of which can support special interface boards.

Lisa's hardware components are organized into convenient plug-in modules. LisaTest™, a diagnostic software program, permits swift determination of any module failures; the plug-in modules allow equally rapid replacement.

Like the dot matrix printer, Apple's daisy-wheel printer produces exactly what is on the user's screen, including graphics. Appropriate for written material that requires letter quality, the printer uses a 130-spoke print wheel and new, multipitch and multifont print wheels. These let the user print a variety of text without changing print wheels. Lisa's daisy-wheel printer prints 45 characters per second.

For greater on-line storage, Lisa can be bolstered with additional five-megabyte ProFile hard disks. These mass storage systems feature faster access and data transfer than traditional floppy disk drives.

Communication Facilities

Using a plug-in modem, LisaTerminal software lets the user gain access to remote mainframe computers, and remote data bases such as CompuServe™, The Source™, and Dow Jones News Service™.

LisaTerminal lets Lisa emulate teletype DEC VT100 or VT52 terminals, providing communications with large DEC-compatible systems. Lisa can communicate with any IBM or IBM-compatible system that supports the 3270 bisynchronous protocols through a 3270BSC software package being developed by Apple.

save!

Prompt delivery and bottom dollar prices make Theta your single source for computer software. Listings represent only a portion of available programs. Call our convenient toll-free number for additional information.

[800] 233-7512 ext. 31
IN PENNSYLVANIA CALL:
[717] 243-5373

Business & Word Processing

ARTSCI			
Magic Window*	79.95	Form Letter Module*	79.95
Magic Mailer*	55.95	Address Book	
Magic Words*	55.95	Mailing List*	39.95
Magic Window II*	119.95	Dataplot*	47.95
IUS		**Omega**	
Easy Writer*	139.95	Locksmith*	79.95
Easy Mailer*	139.95	Inspector*	47.95
Datadex*	119.95	Watson*	39.95
LJK		**Silicon Valley**	
Letter Perfect*	119.95	The Word Handler*	159.95
Data Perfect*	79.95	The List Handler*	71.95
Microsoft		**On-Line**	
Multiplan*	219.95	Screenwriter II Pro*	159.95
muMath/muSimp*	199.95	Screenwriter II*	103.95
Time Manager*	119.95	The Dictionary*	79.95
TASC*	139.95	The General Manager*	119.95
Ramcard*	79.95	Lisa 2.5*	63.95
Softcard*	315.95	**Visicorp**	
Softcard Premium*	599.95	Visicalc*	195.00
Muse		Visiplot/Visitrend*	239.95
Supertext 40/80*	139.95	Visifile*	199.95
Supertext 40-56-70*	99.95	Visipak* (All 3 above)	576.00

Diversions

Choplifter*	27.95	Frogger*	27.95
Starblazer*	25.50	Mouskattack*	27.95
Prisoner II*	26.39	Knights of Diamonds*	27.95
Robot War*	31.95	Beer Run*	23.95
Castle Wolfenstein*	23.95	Bandits*	27.95

Elephant Brand Diskettes* – SS/SD 24.95

NO EXTRA CHARGE FOR MASTERCARD OR VISA. COD ORDERS ADD $3.00. ADD $3.00 FOR SHIPPING & HANDLING (INSIDE USA). PRICES SUBJECT TO CHANGE WITHOUT NOTICE.
*Manufacturer's Trademark

THETA
Research, Inc.
P.O. Box 140 • Carlisle, PA 17013

Circle 72 on Reader Service Card

Now For Rev e!

YOUR KEYS TO . . .

. . . on the Apple][

zardax

Word processing at its finest. Powerful and versatile, yet easy to use and natural. Designed for the business and professional environment (or for anyone who wants the best). Simple control commands. Typewriter-style shift and lock. Glossary. Form letters and mailing lists. Menus for disk access and printing. DOS 3.3 compatible text files. 40 or 80 column display. Modifiable drivers for most interface cards and printers. $295.

ZIP-COMM

Communications add-on for ZARDAX. Turns ZARDAX into a communicating word processor, to send and receive text files. Talk to other Apples, mainframes, information services, typesetters. Includes termina! mode. 300 or 1,200 baud. Works with serial, modem, and popular 80 column cards used by ZARDAX. Log-on files and X-on X-off supported. $80.

AIDA

"Apple Interactive Data Analysis." **Statistical analysis** package for production research work with large survey data files. Full range of analysis — from descriptive statistics to multiple regression. Complete data manipulation, transformation and case selection. Fast and accurate calculations. Up to 4,000 cases and 255 variables per file. $235.

Just push our button . . .

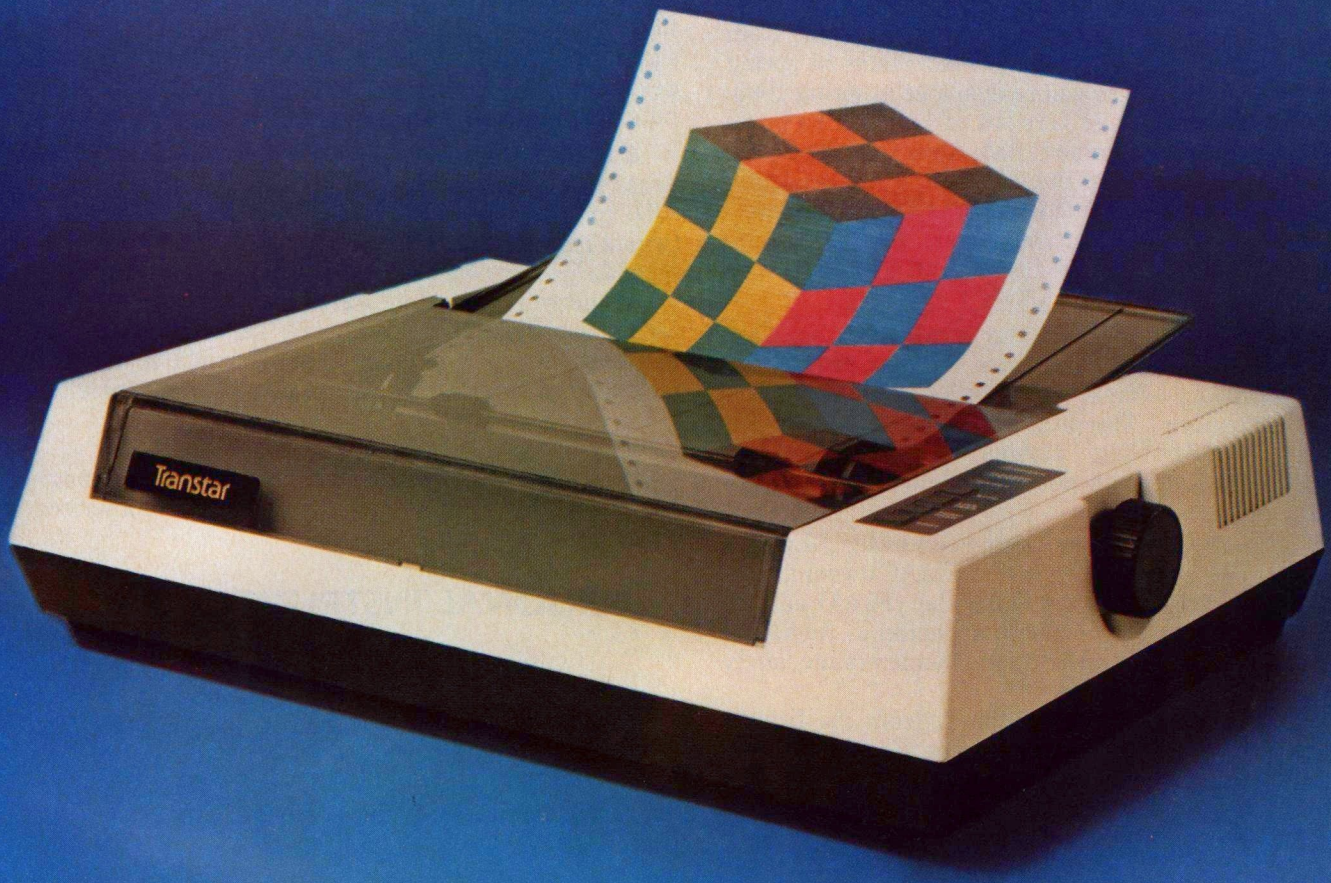
Action-Research Northwest
11442 Marine View Drive, SW.
Seattle, WA 98146
(206) 241-1645 Source: CL2542

Apple][is a trademark of Apple Computer, Inc.
ZARDAX is a trademark of Computer Solutions, Pty., Australia.

Dealer inquiries invited.

Circle 71 on Reader Service Card

Color for the Price of Black and White



Transtar 315

Color Printer for only $599 suggested retail price

- New technology four-hammer print head
- Unique diagonal ribbon provides simpler and more reliable operation
- Prints 7 colors plus more than 30 shades
- All colors and shades are printed in a single pass of the print head
- Built by Seikosha, the oldest manufacturing company of the famous Seiko group—providing precision products since 1892.

Transtar · Box C-96975 Bellevue, Washington 98009

Circle 73 on Reader Service Card

Price And Availability

The Lisa system includes the computer with one megabyte of read/write memory and two built-in floppy disk drives, a five-megabyte ProFile, and the six integrated application programs. This system has a suggested retail price of $9995, and will be available from authorized Lisa dealers in the United States in spring 1983. LisaTerminal also will be available in spring 1983. Several foreign language versions of Lisa will be introduced worldwide in summer 1983. AppleNet, the Ethernet interface device, and the 3270 data communication package will be available in late 1983. Apple Computer, Inc., 20525 Mariani Avenue, Cupertino, CA 95014, (408) 973-3019.

MAESTRO

Vegabondo Enterprises today announced the availability of "Maestro," a companion software product for the "CEEMAC" Visual Composition system for the Apple][computer. The program supports users of the language in assembling "albums" of their own visual creations complete with documentation and necessary runtime modules.

Vegabondo has further announced availability of a low cost commercial license for CEEMAC owners wishing to market visual albums under the "Fire Organ"™ label. The original visual album appeared in August of 1981 and was placed in the public domain immediately thereafter.

Maestro is available immediately from the firm at 1300 E. Algonquin Rd in Schaumburg, IL - 60195 (312) 397-8705, and is priced at $35. Release 1.1 of the CEEMAC language ($75) can be obtained from authorized Apple dealers worldwide or directly from Vagabondo.

APPLE NET

Cupertino, Calif., January 19, 1983—Apple Computer, Inc. has developed a local area network that will allow companies to link together electronically their Apple personal computers to exchange information and share centrally-stored files.

The network, known as AppleNet™, speeds and enhances typical office communications by allowing Apple computer users to electronically transfer information from one system to another quickly and easily. Designed especially to complement Apple personal computer, AppleNet combines reliability and simplicity in design, and is easier to install than a home stereo.

AppleNet links together as few as two systems or can be expanded to meet the needs of a larger office. All Apple computers—Lisa, Apple ///, and Apple][—use AppleNet. Thus, an organization can connect all of its Apples to a single network. At under $500 per connection, AppleNet is a cost-effective solution for both large and small organizations.

AppleNet uses the published and widely-accepted Xerox Network Systems (XNS) Protocols. Apple will encourage the development of network-related products by making public all network interfaces and specifications. All Apple software will be developed to function on both AppleNet and the Xerox/Intel/DEC Ethernet™ standard network.

Network Capabilities

Local area networks enable computers in the same office to communicate with one another via electronic messages sent along a cable. Through AppleNet, users can gain access to information contained in common files and data bases, distribute documents electronically, share peripherals such as expensive mass storage devices and printers, and communicate with other computers and remote data bases.

AppleNet's Baseband Bus Architecture uses Carrier Sense Multiple Access with Collision Detection (CSMA C/D) to avoid data loss during communication. The twinax cable carries information at one million bits per second.

All network electronics are located on an interface card that plugs into the computer. This modular design increases network reliability, and, should a network failure occur, promotes swift isolation of the problem.

Availability

AppleNet will be available in late 1983 through Apple's National Accounts program and authorized Apple dealers.

PERSONAL INSTRUMENTATION

ISAAC 41A

CYBORG has introduced their new economy ISAAC 41A Data Acquisition and Control System which consists of an Apple interface card (with timer, real time clock built in) leading to the 41A unit, which is a 4 slot expansion module enclosure. Also, included is a lab software package; Labsoft extended basic language utilities and application library. Then you select the input/output capability as needed with a variety of ISAAC expansion and remote input modules. With the 41A you buy only what you need, but you can still expand the system by adding new modules in the future. Suitable for both Apple][and *//e*. Northwest Instrument Systems, P.O. Box 2029, Bellevue, WA 98009 (206) 454-3400.

SERIES 401 FFT SPECTRUM ANALYZER

The IQS Series 401 FFT Spectrum Analyzer converts the Apple][and *//e* Computers to a very versatile spectrum analyzer. The Model 401 FFT Spectrum Analyzer is an extremely powerful Apple][based system which allows complete analysis using Fast Fourier Transfer method. This laboratory instrumentation system at $995.00 provides all the tools necessary, both hardware and software to accomplish impulse analysis of physical systems or analyze arbitrary signals for power spectrum, phase or group delay characteristics. Northwest Instrument Systems, P.O. Box 2029, Bellevue, WA 98009 (206) 454-3400.

DIGITAL MEMORY OSCILLOSCOPE

Northwest Instrument Systems' new dual channel, DC to 50 MHz digital memory oscilloscope fits as a module into Apple][and Apple *IIe* microcomputers, using the display and keyboard as an oscilloscope screen and control panel, the Apple computer for waveform processing, and disk memory for waveform storage. The Model 85 performs standard laboratory functions such as signal averaging and DVM readout, and 8K of Apple memory is available for user program development. Price: under $1,000. Northwest Instrument Systems, P.O. Box 2029, Bellevue, WA 98009 (206) 454-3400

ECS COMPUTERIZED GRADEBOOK

Electronic Courseware Systems is pleased to announce the release of a new product entitled, ECS COMPUTERIZED GRADEBOOK, authored by John M. Eddins. This program is designed to assist the classroom teacher in storing, retrieving, and computing grades for students in a systematic manner. Eddins, University of Illinois, at Champaign-Urbana, is an author of a series of published computer lessons, books and articles in the field of computer-based education. The ECS COMPUTERIZED GRADEBOOK will store fifty student names, with up to ten scores per student. The computer program will translate raw scores into standard scores: weight scores by percentage; compute total weighted scores; and assign letter grades. This product is currently available for the Apple][Plus computer (48K, DOS 3.2/3.3). The program is available from Electronic Courseware Systems for $49.95 at the following address: Electronic Courseware Systems, Inc., P.O. Box 2374, Station A, Champaign, IL 61820, (217) 359-7099.

"CP/M® REVEALED"

CP/M Revealed describes in detail the full potential of CP/M — the most popular operating system for microcomputers.

Written by Jack D. Dennon and published by the Hayden Book Company, Inc. of Rochelle Park, New Jersey, the book includes programming exercises that a reader can use with any CP/M based system.

Explained in the book are the technical aspects of CP/M including the console monitor (CCP), the system manager (BDOS), and the input/output driver package (CBIOS). In addition, the data structure of the CP/M disk is fully described.

Detailed discussions on booting up, logging in, changing memory size, mapping disk space, calling programs, file handling and interfacing techniques will be of use to the advanced users.

The book includes CP/M utilities and other essential information for using CP/M effectively. CP/M is a trademark of Digital Research Corporation. CP/M® REVEALED by Jack Dennon, paper, 180 pp., 7 x 10, ISBN-0-8104-5204-9, $13.95. Hayden Book Company, Inc. 50 Essex Street, Rochelle Park, NJ 07662, (201) 843-0550.

FONT GENERATOR ///

FONT GENERATOR /// is a user-oriented program that was designed to incorporate many of the Apple ///'s powerful built-in features to help in the designing of characters FONTS. With it you can test out how your font would look before it is saved.

But that's not all it can do. You can also check out how your font would look in different text modes, and yes, even with different foreground and background colors.

As many as ten character fonts (not including working font) may be loaded at one time. FONT GENERATOR /// also has the capability of copying characters from one font into another one.

Editing and creating characters can be done from either keyboard or joystick (or both). The following characters FONTS are included on the disk:

1) STANDARD
2) APPLE
3) BYTE
4) ROMAN
5) STOP
6) GOTHIC
7) SLANT
8) INVERSE

A character FONT disk is in the making and should become available in a couple of months. All registered owners of FONT GENERATOR /// will be notified by mail when it is available.

If you are interested in purchasing the FONT GENERATOR /// program, please send a check or money order for $40.00 to Apollo Software, 6338 Wisteria Lane, Apollo Beach, Florida, 33570, (813) 645-3153.

CORONA APPLE® II-COMPATIBLE HARD DISKS

Corona Data Systems, Inc., which manufactures and markets the Corona Starfire™ Winchester disk subsystem for the Apple][, today announced that two of the most popular software packages for the Apple][will now run on the Starfire.

The are DB Master™, a database management system from Stoneware Inc. of San Rafael, Calif., and VisiCalc® , the electronic spreadsheet program from VisiCorp, which requires The Transporter™, from Omega Microware, of Chicago, Ill., a utility for moving a purchased copy of VisiCalc onto hard disk.

As described by a Stoneware spokesman, DB Master is a sophisticated database management applications package incorporating powerful report generation facilities and versatile screen formatting commands. Multiple primary and secondary search keys are easily designed by the first time user of DB Master.

The hard disk version of DB Master, which retails for $449 and is described by its manufacturer as the most powerful database manager produced for the Apple][, provides hard disk users with greater flexibility and capacity and faster access than is available from other similar products, according to Stoneware's spokesman.

An optional product—DB Master Utility Pak#1 (retail price $99)—provides users with four of the most popular supplementary programs currently available on the software market. These are capabilities for translation, restructuring, recovery of damaged files and file reproduction.

The Transporter, which retails for $49.95, is a utility designed to move a purchased copy of VisiCalc onto a Corona Starfire hard disk in order to allow the use of VisiCalc in the hard disk environment.

An enhanced version of The Transporter, which will be available in February 1983 and retail for $125, will support systems with 80-column card and a RAMEX 128™ memory expansion card. This will expand the VisiCalc workspace to 127K bytes of memory and add additional features to VisiCalc similar to those found on advanced versions of the software, according to Alpert.

Corona does not support and cannot guarantee availability of these software products. Additional information about them is available directly through the companies that manufacture and market them, or from local microcomputer dealers. Dealer inquiries to the companies are invited. Corona, 31324 Via Colinas, Suite 110, Westlake Village, CA 91361, (213) 706-1505.

ROADSEARCH

Columbia Software (Columbia, MD) has introduced a computerized Road Atlas for the Apple][personal computer. Called ROADSEARCH, this program simplifies the tedious process of determining driving routes, mileages, travel times, and fuel usage.

ROADSEARCH develops different types of routes. The most powerful program finds the shortest practical route between cities in its database. This program also avoids toll or other roads if desired. Another program developes routes which may be longer, but more suitable to a user's specific needs.

ROADSEARCH also contains a unique subroutine that estimates flying time between cities.

ROADSEARCH has a database of 406 cities and road intersections located in the USA and Canada. Also included in the database are about 69,000 miles of interstate and major through highways. If necessary, this database can be modified with updated road mileages.

The printed outputs from ROADSEARCH are an excellent companion on any trip. They include the driving route, distances, travel times, and fuel usage tailored to vehicle average miles per gallon (MPG).

ROADSEARCH has been human engineered and extensively tested to assure reliability. Error trapping is employed throughout to help avoid most errors. There should be little difficulty mastering ROADSEARCH in a very short time.

Columbia Software is continuing its policy of delivering unlocked software. This means that back-up copies can be made with most standard copy programs.

ROADSEARCH requires an Apple][personal computer with DOS 3.3. Price is $34.95. For more information contact Columbia Software, P.O. Box 2235, Columbia Maryland 21045 (301) 997-3100.

MASA

The Apple][microcomputer offers rapid synthetic seismogram generation and manipulation through the release of a new geophysical software package offered by Sperling Geoscience DataCorp. The software package named MASA, Micro Assisted Seismic Analysis, provides easy "user friendly" digitization of sonic logs and plots industry standard (7.5 in/sec) synthetic seismograms.

For interpretive work, a 300 ms synthetic seismogram window, taken from a digitized sonic log, can also be generated in less than one minute with a subsequent hard copy produced in one minute thereafter. Editing of the sonic log data (i.e. unit thickness and/or sonic velocity) can be accomplished by using the very "user friendly" insert, change, delete, and add modes provided. This allows the Geophysicist to create any geological relationship and/or stratigraphy he desires and in less than 2 minutes have, in hand, a synthetic seismogram representative of that geological setting. Synthetic seismograms can also be placed side by side to form a synthetic seismic section to provide a two-dimensional seismic model of the corresponding geological scenario.

MASA also includes a complete set of accessory programs for the geophysicist, such as: Time to depth conversion; RMS velocity function determination from sonic logs; Incident ray tracing; Source/Receiver vs Depth ray tracing; Sonic log expansion and compression; Append sonic logs; And, several utility programs.

For more information, contact Sperling Geoscience DataCorp, 600 Bluebell Drive, Lansing, Michigan 48910, (517) 393-1296.

IBM OS/JCL

Applied MicroSystems, Inc., announces the release of their newest educational package, Introduction To IBM OS/JCL. Intro. To JCL operates on an Apple][with Applesoft and DOS 3.3. This interactive tutorial is aimed at individuals who work in IBM OS batch environments. Ten lessons teach OS concepts, JCL syntax and usage, and introduce the student to the IBM utilities.

In addition to the normal introductory JCL material, Introduction To IBM OS/JCL covers spooling, blocking, cataloged data sets, condition codes, cataloged procedures, partitioned data sets, and storage allocation. A unique lesson format allows the student to advance or back up thru the lessons at his own pace. An institutional version of the course, which keeps track of up to 100 students, is available.

Price:	Individual version	– $25
	Institutional version	– $50

For more information phone: (404) 475–0832.

18SRC (STATIC RAM CARD)

Pontiac, MI, Legend Industries, Ltd. has developed the first, as far as we know, 18K Static RAM card for the Apple][,][+ and *IIe* computers. The card contains a battery back up system to allow memory retension after power–down, enabling the user to store programs and/or special monitor routines or for that matter, DOS; if the user desires.

The card draws very low power and can be used in slot. A write–enable/write–protect switch allows the user to protect the information in the card or alter the data at will. This is great for those monitor routines that are not alterable in the ROM on the motherboard, such as CAPTEST. One other application is to relocate DOS into the card and install a special "BOOT" routine that would automatically install DOS into the machine. This routine can be installed in the C800–CFFF space provided by Legend on the RAM card.

The LEGEND 18SRC is another useful product from Legend Industries or your local dealer. Legend Industries, Ltd., 2220 Scott Lake Road, Pontiac, MI 48054, Editor's Contact: Lynne McLaughlin.

NEW DATA SHEET DESCRIBES AMDEK CORPORATIONS'S NEW 3" MICRO-FLOPPYDISK DRIVE SYSTEM

The new 3" Micro–Floppydisk Drive System for personal and business computers is highlighted in a 4–color, two–page data sheet from Amdek Corporation, Elk Grove Village, Illinois.

The compact Micro–Floppydisk Drive accommodates two cartridges for up to one (1) megabyte capacity. The system is plug compatible with standard 5¼" floppydisk drives in recording format, data transfer rate and disk rotation speed.

The system includes two drives (with built–in power supply) for the unique "shirt–pocket" size Micro–Floppydisk Cartridges. The hard plastic cartridge has a flip–type head cover to protect it from dust, scratches, or finger prints.

This informative data sheet is available by contacting: Amdek Corporation, 2201 Lively Blvd., Elk Grove Village, Illinois 60007. Telephone: (312) 364–1180. TLX: 25–4786.

FORTRAN 77

Softronics, Inc. has released a FORTRAN 77 source program which provides Softrans protocol compatibility for file transfers between Apple personal computers and *any* host computer.

Softrans is used in conjunction with Softerm™, the company's high speed CRT terminal emulation program for the Apple computer, to exchange data between the Apple Computer and any host computer which provides FORTRAN 77 language capability. FORTRAN 77 is an industry standard high–level language available on most computer systems.

The Softrans protocol, which operates in a block mode using asynchronous communications, provides error detection with automatic retransmission, automatic binary encoding and decoding, CRC–16 checksum, and data compression to enhance line utilization. The FORTRAN 77 program is executed on the host system under the control of the Apple computer which when running the Softerm program appears to the host computer as a standard CRT terminal. Commands may be entered which allow file directories to be displayed and files to be transferred to and from the Apple and host computer system.

Softerm file transfer utilizes an easy–to–use command language which allows simple definition of even complex multiple–file transfers with handshaking. Twenty–three high level commands are included which may be executed interactively or from a file transfer macro command file which has been previously entered and saved on disk.

The Softerm terminal emulation program including a 300 page user's guide, terminal emulations for the IBM 3101, DEC VT102, and VT52, Data General D200, Datapoint 3601, ADDS Regent 20, 25, and 40, ADDS Viewpoint, Lear Siegler ADM–3A and ADM–5, Hazeltine 1400, 1410, and 1500, Teletype Model 33 KSR, and TeleVideo 910, and the Softrans protocol and FORTRAN 77 source program, is available for $150 from Softronics, Inc., 6626 Prince Edward Place, Memphis, TN 38139, (901) 755–5006.

RENTAL OUTLET OPENS

Monclair, NJ (January 11, 1983) — The nation's first rental outlet for personal computer software opened here today.

The concept of the Software Library is to put the latest software in people's hands at substantially reduced costs. It also is designed to protect buyers from software obsolescence and assure their satisfaction with a program's capabilities.

Renting keeps the investment in software minimal. Rental fees at the software library are usually less than ¼ to ⅓ of the list price. By renting instead of purchasing, the user doesn't have to hold on to game and entertainment programs in which interest has waned. Furthermore, parents will not have to worry about buying an educational program that will soon lose its value because their children have mastered it.

At The Software Library, users rent the software for a week or longer, depending upon the program. If they decide to purchase the software, the rental cost is applied fully toward the purchase price.

Business software also will be rented. Too frequently, expensive programs must be purchased without providing users an opportunity to test software in their own business environment. Glancing through documentation, or brief demonstrations of complicated word processing or data base management systems do not always assure the business user that the program will suit his or her needs — or the needs of the staff.

Computer hardware and peripherals also will be available at The Software Library which is located at 700 Bloomfield Avenue in Montclair, NJ. (201) 783-6412.

BIT 3 COMPUTER CORPORATION INTRODUCES UNIVERSAL MATING DEVICE FOR RS232 CONNECTORS

Minneapolis — Bit 3 COMPUTER CORPORATION today introduced a RS232 Connector Matrix that permits the mating of *any* two RS232 connectors. The RS232 Connector Matrix eliminated the need to fabricate special cables or resolder existing cable wiring to achieve a RS232 interface between two equipments.

It is an invaluable tool for

- cable pin number crossconnections
- gender connector changes (male or female or vice versa)
- diagnostic troubleshooting

Some typical applications are

- Use as a NULL-MODEM
- Converter for DCE/DTE pin conventions
- Diagnostic line monitor

The product consists of a 10x10 position miniature matrix slide switch, four RS232 connectors (two male and two female), and nine LEDs, all of which are mounted on a 3.5 inch square printed circuit board. The 10x10 matrix switch permits crossconnecting pins 2,3,4, 5,6,8,11,19 and 20 of any two of the four RS232 connectors. The nine LEDs monitor RxD, TxD, RTS, CTS, DTR, DSR, DCD, as well as pins 11 and 19. The product therefore permits RS232 cables of any connector gender combination to be used to connect equipments together. Batteries or other external power sources are not required.

Many printers are now designed both for use with a modem or for direct connection to a computer. When connected to a modem the printer normally uses the xON/xOFF protocol to inform the remote computer that it can accept more data to print. If directly connected to the computer the computer expects the printer to present a signal on pin 5 (CTS) indicating that the printer can accept more data. Generally this signal can be found on pin 11 (Centronics, Okidata) or on pin 19 (NEC). The Bit 3 RS232 Connector Matrix permits the necessary changes without building a special cable.

Bit 3 manufactures products for the computer industry. Price is $89.00 FOB Minneapolis. Availability:, stock. (612) 881-6955.

APPLE — NCS ANNOUNCE SCHOOL SOFTWARE GIVEAWAY

Apple Computer Company, the nation's leading classroom computer manufacturer, and National Community Services, a nationwide fundraising company, have announced a cooperative program to donate a valuable collection of more than ten computer software demonstration disks to any accredited public or private elementary or secondary schools in the nation. The schools have only to purchase Apple II Plus personal computers financed through an NCS fundraising program.

The free demonstration disks have been reviewed by leading computer educators, and are generally regarded to be among the finest available. The free disks included in the Apple-NCS premium offer are:

1. Edu-ware Services, Inc: math programs from level two through adult
2. J & S Software: biology, physics and chemistry grade books for levels seven to twelve
3. The Learning Company: learning programs for the pre-school and the elementary level
4. Reader's Digest Education Division: math, level one to seven; problem solving, level five to eight
5. Universal Systems for Education: a range of problems for grade levels two to six
6. Milton Bradley Company: math and language discs, levels six to eight
7. Versa Computing: program demos featuring alphabet, colors, solar system, U.S. presidents and the globe, levels pre-school through high school
8. Window, Inc.: math, level K-8; entry-level programming and advanced programming, levels 9-12
9. Educational Activities, Inc.: language, reading, math, and science, levels four to six; remedial, levels seven and eight
10. Lightning Software: typing demo disk, levels K-12
11. Broderbund Software: math programs from level two through adult

National Community Services offers a broad range of fundraising programs that are conducted on behalf of schools, school groups, and community organizations. The programs, directed by former professional educators representing NCS, include product sales (magazine subscriptions, posters) as well as fundraising efforts that require no sales or deliveries whatsoever.

The joint venture with Apple is the latest NCS effort to assist schools in developing financial resources they would otherwise not have.

For information on this software offer and fundraising possibilities, call Nadine Vantine at 800-225-7934, or write to her c/o NCS, 217 Jackson Street, Lowell, Massachusetts 01853.

CONSULTANTS

Bob Huelsdonk	Apple ///(Western U.S.)	(206) 362-4910
Stuart Yusem	Apple /// (Eastern U.S.)	(609) 822-9306
Erv Edge	Assembly (Alaska)	(907)337-7620
Dennis Baumgarte	Assembly (Eastern U.S.)	(914) 473-3210
Don Williams	Assembly (Western U.S.)	(206) 242-6807
Michael Thyng	Business	(206) 524-2744
Steven E. Bair	Data Bases	(913) 782-5901
William P. Steinberg	DOS (Eastern U.S.)	(212) 921-0663
Murray Arnow	CPM/Engineering Application	(312) 291-9210
Art Schumer	DOS (Northern U.S.)	(701) 282-7907
Lee Meador	DOS (Southern U.S.)	(817) 469-6019
Kurt Thams	DOS/Gen'l (Western U.S.)	(415) 327-2116
WM. Reynolds III	DOS/Tech (Western U.S.)	(503) 472-4028
Ted Perry	Education	(916) 485-1690
Cornelis Bongers	European Consultant	010-525511 Ext. 3275
Alan Robinson	Fortran	(503) 757-8234
Jim Eatherly	Games	(202) 232-6046
Tedd Sperling	Geoscience	(517) 393-1296
Stan Axelrod	Hardware Design/I-O Control	(415) 488-0915
Tom Deal, M.D.	Handicapped Applications	(206) 634-2847
Randy Halvorson	Hardware Interfacing	(206) 783-9040
Ed So	Hi-Res	(408) 267-3448
Stowe Keller	Mathematics	(804) 973-5701
Fred Merchant, M.D.	Medical Group	(415) 921-7792
Stephen F. Lee	Music	(704) 289-4102
Tom Woteki	Pascal (Eastern U.S.)	(202) 547-0984
Randy Warrens	Pilot Language	(608) 837-9566
Hersch Pilloff	Stock Market	(301) 292-3100
Mark Robbins	Telecommunications	(303) 755-6440
Jim Hockenhull	Video Interfacing	(509) 334-4137
George L. Hefter	VisiCalc (East)	(617) 256-8148
Jerry Brieger	VisiCalc (West)	(206) 881-1241
Vince Aderente	VisiCalc (North)	(312) 289-9685
Dr. Bob Richard	VisiCalc (South)	(504) 466-0894
Brian Abrahams	VisiCalc (South Africa)	(Durban) 315442

AD INDEX

A.P.P.L.E. 86, 89, 100, 105, 113, 117
Action-Research Northwest 120
Anthro-Digital Inc. 58
Apple Computer Inc. 64, 65
Applied Analytics 63
Applied Microsystems 117
Assembler Teacher 12
Balbes Software Systems 45
Beagle Brothers 36
Beaman Porter 92
Bytes & Pieces 110
Columbia Software 106
Computer Case Co. 110
Dajax 116
Datamost 35
David - Data 7
Deltrex 8
Diversified Software Research 61
Expotek 110
Frobco 117
Futra Company 20
Gourmet Software 54
Gravymakers 62
IJG, Inc. 6
Interactive Microwave, Inc. 106
Ivers Specialties 98
Johnson Associates 108
Laing Electronics Inc. 86
Leading Edge Products ..... IFC, BC
Legend Industries, Ltd. 54
Micro Max 35
Mind Systems Corp. 90
Northeast Expositions 56
OHM/Electronics 4
Omega Microwave 50
Orange Micro 72, 73
Personal Computer Products 80
Pirates Harbor 74
Professional Data Systems 88
Proforma Software 98
Proline Electronics 75
R.C. Electronics 112
R.H. Electronics 1, 87
Scientific Software Products 23
Sensible Software 70
Sof/Sys, Inc. 55, 103
Soft CTRL Systems 102
Soft Key Publishing 102
Softronics 49
The Software Source 91
Southwestern Data Systems 97
SSM Microcomputer Products 106
Star Logic 90
Stellation II 12, 98
Super Software Systems 97
Tava Corp. 104
Telephone Software Connection 117
Texas Ranch & Shoreline Systems 66
Theta Research, Inc. 120
Townsend Microware 108
Transtar 46, 121
United Computer 19
Vista Computer Co., Inc. ..... IBC
XPS, Inc. 108

A.P.P.L.E. ORDER FORM

DATE _____

MEMBERSHIP NUMBER _____ EXPIRATION DATE _____

NAME _____ ADDRESS _____

CITY _____ STATE _____ ZIP _____ PHONE _____

PRODUCT	UNIT PRICE	*FOREIGN SHIPPING CHARGES (See below)	AMOUNT

FOREIGN SHIPPING/HANDLING CHARGES CHART

In an attempt to simplify things for our members outside the United States, we have devised this easy rate chart for figuring the shipping and handling charges on your order. Simply determine the **TOTAL** amount of your order and add the correct amount as corresponds to the rate chart. (NOTE: THIS DOES NOT APPLY TO APPLICATIONS OR RENEWALS — BUT TO PRODUCT ORDERS ONLY!)

total amount of order	shipping/handling
$ 0.00 - $ 20.00	add $ 6.00 to order
$ 20.01 - $ 60.00	add $12.00 to order
$ 60.01 - $100.00	add $18.00 to order
$100.01 - $140.00	add $24.00 to order
$140.01 - $175.00	add $30.00 to order
$175.01 - and over	add $36.00 to order

Note to Foreign Members: All hardware products, except APPLCASE lower case adapters, must include a United States address. These products cannot be shipped outside U.S.

VISA Account No. _____ Exp. Date _____

MasterCard Account No. _____ Exp. Date _____

Subtotal _____

**Washington sales tax _____

TOTAL AMOUNT _____

* Foreign Members Only

** Washington state residents add 6.8% state sales tax.

SEND TO: A.P.P.L.E.
21246 68th Ave. S.
Kent, WA 98032
(206) 872-2245

***NOTE: Canadian shipping/handling chart is being revised.**

REQUEST FOR CHANGE OF ADDRESS

Dear A.P.P.L.E.,

I am moving and my present address is:

PRESENT:

NAME _____

ADDRESS _____

CITY _____

STATE _____ ZIP CODE _____

EFFECTIVE DATE _____

NEW ADDRESS:

NAME _____

ADDRESS _____

CITY _____

STATE _____ ZIP CODE _____

MEMBERSHIP NUMBER _____

*Please fill out and return this notice two weeks before moving.

UNCLASSIFIEDS — A.P.P.L.E ORDERS

CALL -A.P.P.L.E. (un)Classifieds

Members **Non-commerical** rate .05/word
Members commerical rate .25/word
All others: 1.00/word, $25 minimum
Send ad copy with payment to editorial office

APPLE SOFTWARE RENTAL & sales. Over 2500 programs available. Rent that expensive software for just pennies a day! Digital Delli Software. 2961 S. Las Palmas, Mesa, AZ 85202 (602) 820-0082.

ANNOUNCING ROADSEARCH for your trips. This powerful program computes the shortest practical auto routes between cities in the USA/Canada. . .and more! Database contains 406 cities and 69,000 road miles. Outputs include route, mileage, time and gallons. Printer optional. Ten day moneyback GUARANTEE. Only $34.95 (plus $1.50 s/h). Check/VISA/MC. Columbia Software, Box 2235K, Columbia, MD 21045. (301) 997-3100.

SEND FOR YOUR FREE CATALOG FROM TECK ASSOCIATES — producers of top quality computer assisted instructional materials in BASIC SKILLS, AUTHORING, SCIENCE and EARLY LEARNING. Contact TECK Associates, Box 8732, Dept. CA, White Bear Lake, MN 55110.

FOR SALE: CHATSWORTH OMR-500 with Apple II computer interface (others possible). New, never used due to change in requirements. — Original price $1,050. — Selling price $500./ or highest bidder. MICROMODEM II, (brand new, barely used) - $250. APPLEWRITER II, (brand new, never opened) - $75. SUPERKRAM/CONQUEST 0.5 - Relational data base system (ROM board, original discs and manuals). - $150.00. Barry Gordon, M.D., Ph. D., 4340 Roland Springs Drive, Baltimore, Maryland 21210. (301) 955-6431 (daytime; (301) 235-3125 (evenings).

APPLE PASCAL ANIMATION GRAPHICS Morwe/Graphics - 83 pp. manual and two diskettes. SASE for details, $3.95 for manual. ★★-$19.95-★★ for complete kit. HENDERSON ASSOCIATES. 980 Henderson Ave., Sunnyvale, CA 94086.

RENT SOFTWARE $5 month or buy it 50% discount or sell it or receive credit on purchases or rentals or swap it. Call (206) 784-2081 or send SASE to: Preowned Software, 4229 Baker Ave. N.W., Seattle WA 98107 for details.

SOFTWARE LESS THAN HALF PRICE! All used, but ORIGINAL DISKS WITH ORIGINAL DOCUMENTATION. DISK GAMES: Zero Gravity Pinball, Hi-Res Adventure #1 and #2, Blisterball/Mad Bomber, A2-FS1 Flight Simulator, Space Raiders, Asteroids, Tranquility Base, Horizon V. CASSETTE GAMES: Space Tilt, Fastgammon (backgammon). NON-GAME: Eliza 3.0, DOS Boss, Appletalker, Typing Tutor (tape). All disks priced $10 to $15, tapes $5 to $7. Might consider trades. Also have Magic Window I WORD PROCESSOR for only $40 ($100 new). HARDWARE: Vista Model 150 40 character type ahead buffer for $20 (half price). Kevin Talbot, 8056 SE 72nd St., Mercer Island, WA 98040; (206) 232-7058 SY, (206) 643-9349 EVE.

TIME ZONE HINT BOOK. Are you tired of being frustrated by playing the Adventure, Time Zone? Book supplies all needed answers to solve game. Send $5.00 to: Hint Book, 5419 Johannsen Ave., Dayton, Ohio 45424.

MAGAZINE ON DISKETTE - SOFTDISK is published monthly on a two-sided diskette. It is chock-full of Apple news, hints, tips and loaded with unprotected programs ready-to-run. SOFT DISK is just $5 per month when you return the previous diskette. To start, send $10 (or $5 for A.P.P.L.E. members with your membership number) to: SOFTDISK MAGAZETTE, Dept. A1, 3811 St. Vincent, Shreveport, Louisiana 71108.

INTERESTED IN APPLE PROGRAMS ON CASSETTE, INT or FP. List & Price to: W. S. Skeen, Rt. 1, Ager Rd., Hornbrook, CA 96044.

AIR NAVIGATION TRAINER features VOR's, NOB's, ADF, adjustable winds, ground-track map and more. Call or write for more info. SPACE-TIME ASSOCIATES, 20-39 Country Club Drive, Manchester, NH 03102, (603) 625-1094, MC and VISA, CL0393 and 70310,217 US$40.

GETTING MARRIED - MUST SELL. Apple parallel card - $100, Lisa 2.5, The Dictionary - $40, Alpha Plot - $20, Hires Adve. No. 2, Apple Panic, Meteoroids in Space, Apple-Cillin, Space Eggs, Choplifter, Super Disk Copy III - $15. Norm Pennington, 1426 John Glenn Rd. #E, Dayton, Ohio 45410. (513) 254-6854 evenings.

CARL WINTER: URGENT!
Important you call the Hotline (206) 932-6588.

CALL —A.P.P.L.E. CONSULTANTS WANTED, particularly in following fields: Pascal, Apple ///, Business, Data Bases, Hi-Res, Forth, CP/M, Logo, other languages and applications. Please send a brief statement of qualifications to the Editorial Dept., 5606 Calif. Av. SW., Seattle, WA 98136, including telephone number to be used and hours available. Our consultants are unpaid volunteers who wish to assist other members.

STOCK OPTION INVESTORS: Call/Put Options 2.0. New program calculates call and put prices, hedge ratios, break-even prices, % change per dollar stock change, and future price prediction. Requires Apple™ II plus, 48K, DOS 3.3, single disk or Apple™ III in emmulation mode. Send check to Harloff Inc., Software Dept. CA, 725 Market St., Wilmington, Delaware 19801. Only $150.00

ETI INTERFACE turns your Centronics 737 into an intelligent printer: automatic proportional spacing, justification, centering etc. plugs between any parallel card and 737, no modifications. As new, US $100 — MACROTRONICS A650, an excellent CW/RTTY hardware in cabinet, with software including CW training mode, automatic message insertion, much more! As new, US $200 — SUP'R' TERM 80 columns videoboard, last revision, as new, US $200. Send check to PAT DUBOIS, 7720 Malahat Ave, Richmond, B.C. V7A4H2, Canada.

★★ TV GAMES DESIGNERS ★★ Use your Apple and our VCS interface card to write your own programs for the explosive TV game cartridge market. For free brochure write: GAME DEVELOPMENT CORP., Box 654, Gilbert, Ariz. 85234.

ALMOST WHOLESALE — Rana Elite I drive $299. Epson MXBO/FT $489; FX80/FT $599; MX100 $679. ALS new Z80b & CP/M card $299. IDS Prism 132 $1599. Multi-plan (Apple DOS) $179. Add 3% shipping/handling. VISA/MC add 4% (include exp. date). Please INCLUDE PHONE NUMBER WITH ALL ORDERS. For fast delivery, send cashier's or certified check. Washington residents include 6.2% sales tax. COMPUTER PRODUCTS UNLIMITED. 11116 East 46th Ave., Spokane, WA 99206. (509) 922-1346 (eves.).

HOW TO EFFECTIVELY MARKET YOUR COMPUTER PROGRAMS. Avoid pitfalls that can cost you money. Copyright procedures — Marketing on cassette and disk — License requirements — Starting a software business — Selling to software companies — MORE! $19.95 postpaid. Celestial Software, 3010 Warrington Ave., Lakeland, FLA. 33803.

HUMOROUS BOOKS FOR COMPUTER PEOPLE. Read about computing projects that failed, computing companies that failed, unusual people in computing. Free brochure. COMPUTING TRENDS, 6925 - 56th Ave. S., Seattle, WA 98118.

UPGRADE FROM INTEGER TO APPLESOFT in ROM with Five (#341-0011 to #341-0015) Almost new chips. $47. Write to: Bruce Norgan, N. 11618 Madison, Spokane, WA 99218.

PRODUCTS NOW AVAILABLE TO A.P.P.L.E. MEMBERS

Call —A.P.P.L.E. Specials

All About Eamon[1]	$95.00
Eamon Special[2]	$35.00
All About DOS[2] Intro Price	$12.50

A.P.P.L.E. LIBRARY

"Apple Orchard" No. 1 (magazine)	$2.50
Apple Users Guide	$13.00
Assembly Lines: The Book	$16.50
Beneath the Apple DOS	$17.00
"Best of Cider Press" (S.F. Newsletter Annual) Vol. 1	$4.50
Call A.P.P.L.E. Annuals	
1978	$7.50
1979	$10.00
1980	$15.00
1981	$18.50
1982	$18.50
Call —A.P.P.L.E. in Depth	
1: All About Applesoft	$17.00
2: All About Pascal	$11.00
3: All About DOS	$13.50
Kids and the Apple	$16.00
Power of VisiCalc Vol 1 or II, each	$8.50
Using 6502 Assembly Language	$17.00
Woz Talks (VHS Video cassette)	$27.50

AS IS SOFTWARE

Public domain diskettes (DOS 3.2.1) (Send 37¢ S.A.S.E. for list of titles)	ea. $4.00
Eamon Adventure Diskettes	ea. $4.00

A.P.P.L.E. DISKPAKS

Call —A.P.P.L.E. Anthology Vols I-VII, incl.ea.	$12.50
DICE 82[7]	$12.50
Diskpak 12	$16.50
Diskpak 34A	$13.00
Diskpak 5	$13.00
Diskpak 6B (DOS 3.2 disk utility)	$18.50
Diskpak 6B-16 (DOS 3.3 disk utility)	$18.50
Diskpak 7	$17.50
Diskpak 8	$13.00
In Depth 1 Diskette	$16.50
In Depth 2 Diskette Set	$20.00
Pascal Anthology '81	$12.50

A.P.P.L.E. HARDWARE

ApController	$95.00
Ap Coolkit	$22.50
AP DRIVE	$325.00
AP DRIVE with Controller	$420.00
Ap L Case 1	$30.00
Ap L Case 2 (Lower case chip)	$27.50
Disk Banks	2 for $9.00

AP• Jewelry

Lapel pin	$3.00

A.P.P.L.E. PRESTIGE Game Diskettes

Cosmic Combat	$12.50
Mummy's Curse	$12.50

Apple Prestige Product LinE

APfile — Text file utility	$18.50
ApGraphics: The Filler	$22.00
Applesoft Carpenter	$18.50
Aptest	$18.50
Big Mac Macro-Assembler/Ted	$22.50
Big Mac LC	$28.50
Global Program Line Editor	$38.50
Higher Fonts 1	$13.00
Higher Text Plus	$27.50
Program Global Editor	$18.50
Program Line Editor	$20.00
Soft Seventy	$22.00
Symbol Symon	$20.00
WordWeaver ///	$75.00

[1] Expires April 30, 1983
[2] Expires April 30, 1983
[7] $7.50 With Ap L Case 1
$10.00 with Ap L Case 2

Diskettes supplied on DOS 3.3 except as noted.
All walkin orders must be called in prior to pickup.

NOTICE TO FOREIGN MEMBERS:
A.P.P.L.E. Hardware products not available to overseas or foreign members. This includes Ap.Drive & Ap Cool Kit.

Washington residents add 6.8% sales tax

ORDER FORM ON REVERSE

VistaDaVinci
GREAT INNOVATORS

15th century renaissance man, Leonardo Da Vinci, contributed an inexhaustible collection of inventions and ideas to solve the world's problems. These ideas were centuries ahead of their time. The studies on this page deal with experiments in manned flight.

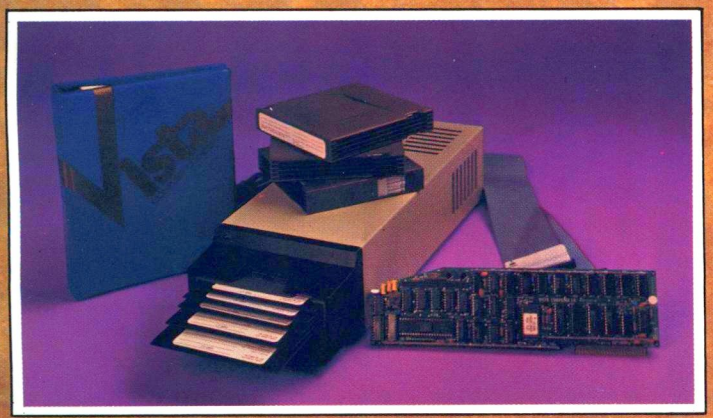
Like Leonardo, Vista Computer is answering many of today's complex computer storage problems. The Vista V1200 is a great solution to Apple II™ storage. Mass storage for your Apple II™ Computer has always been a problem. On one hand, there were the exotic, expensive hard disks with no cost efficient means of backup. On the other hand, the Apple floppy drive lacked the speed and storage demanded by today's professionals.

Vista's V1200 offers both at an incredibly attractive price. The removable VistaPak cartridges offer 6 Megebytes of removable storage each and can be backed up like a floppy.

Now hard disk storage and speed can be yours with the added capability of interchangeable media. The V1200 eliminates the worries of head crashes, drive alignments, lost data, or backup with a new application of field-proven floppy technology.

The VistaPak cartridges hold 6MB of formatted data each. The removable cartridge allows you to keep duplicates of your valuable data as well as to keep separate paks for your accounting, word processing, spread sheet and other applications. No other storage device offers more in flexibility and capability.

• Microprocessor controlled drive • DMA Controller • Removable Data Cartridges • CP/M, DOS & Pascal compatible • Quickcharge™, DOS enhancement included • Includes 1 VistaPak cartridge • Vista 120 Day Warranty



Vista V1200

™Apple II is a registered trademark of Apple Computer Co.

Contact Your Local Vista Dealer or Call our Vista Hotlines.

Vista COMPUTER COMPANY, INC.

1317 East Edinger / Santa Ana, CA 92705
(714) 953-0523 / (800) 854-8017

DISTRIBUTORS / REPRESENTATIVES

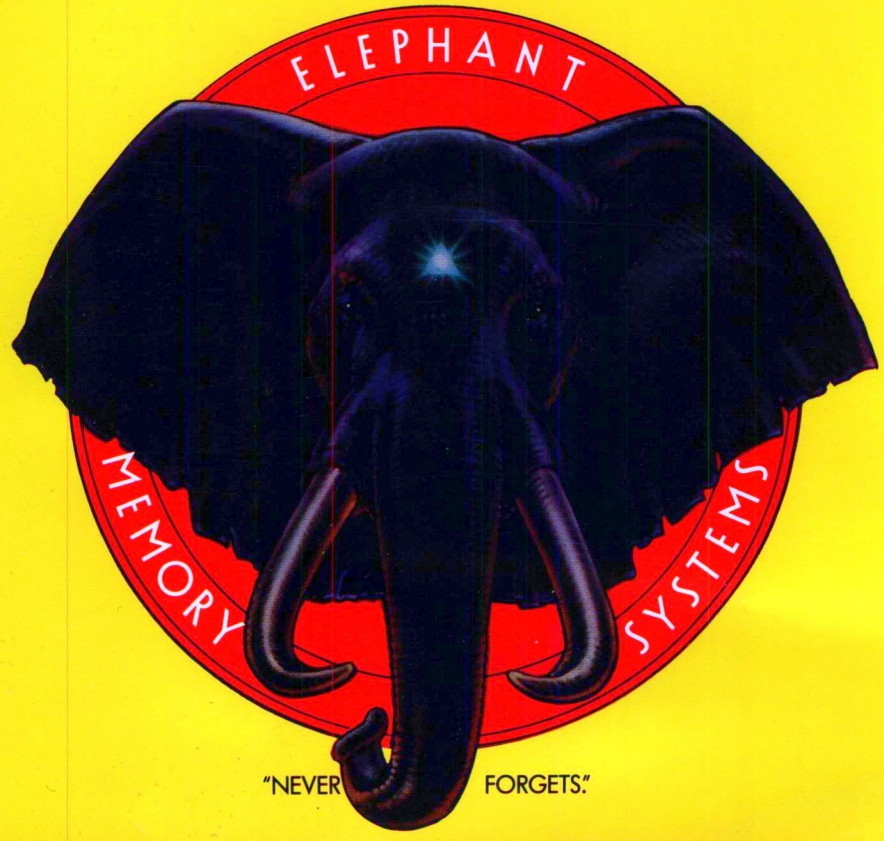
Western - Group 3 Wholesale
(213) 973-7844 (408) 732-1307

South Central - M.P. Systems
(214) 385-8885

Northeast - Computers & Peripherals Inc.
(315) 476-6664

Circle 66 on Reader Service Card

REMEMBER:



MORE THAN JUST ANOTHER PRETTY FACE.

Says who? Says ANSI.

Specifically, subcommittee X3B8 of the American National Standards Institute (ANSI) says so. The fact is all Elephant™ floppies meet or exceed the specs required to meet or exceed all their standards.

But just who is "subcommittee X3B8" to issue such pronouncements?

They're a group of people representing a large, well-balanced cross section of disciplines—from academia, government agencies, and the computer industry. People from places like IBM, Hewlett-Packard, 3M, Lawrence Livermore Labs, The U.S. Department of Defense, Honeywell and The Association of Computer Programmers and Analysts. In short, it's a bunch of high-caliber nitpickers whose mission, it seems, in order to make better disks for consumers, is also to make life miserable for everyone in the disk-making business.

How? By gathering together periodically (often, one suspects, under the full moon) to concoct more and more rules to increase the quality of flexible disks. Their most recent rule book runs over 20 single-spaced pages—listing, and insisting upon—hundreds upon hundreds of standards a disk must meet in order to be blessed by ANSI. (And thereby be taken seriously by people who take disks seriously.)

In fact, if you'd like a copy of this formidable document, for free, just let us know and we'll send you one. Because once you know what it takes to make an Elephant for ANSI . . .

We think you'll want us to make some Elephants for you.

ELEPHANT.™ HEAVY DUTY DISKS.

For a free poster-size portrait of our powerful pachyderm, please write us.
Distributed Exclusively by **Leading Edge Products, Inc.**, 225 Turnpike Street, Canton, Massachusetts 02021
Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.

Circle 76 on Reader Service Card